VT50/52/55

VT50A,B,H/52 ACCEPTANCE TEST

MD-11-DZVTC-E

EP-DZVTC-E-DL-E

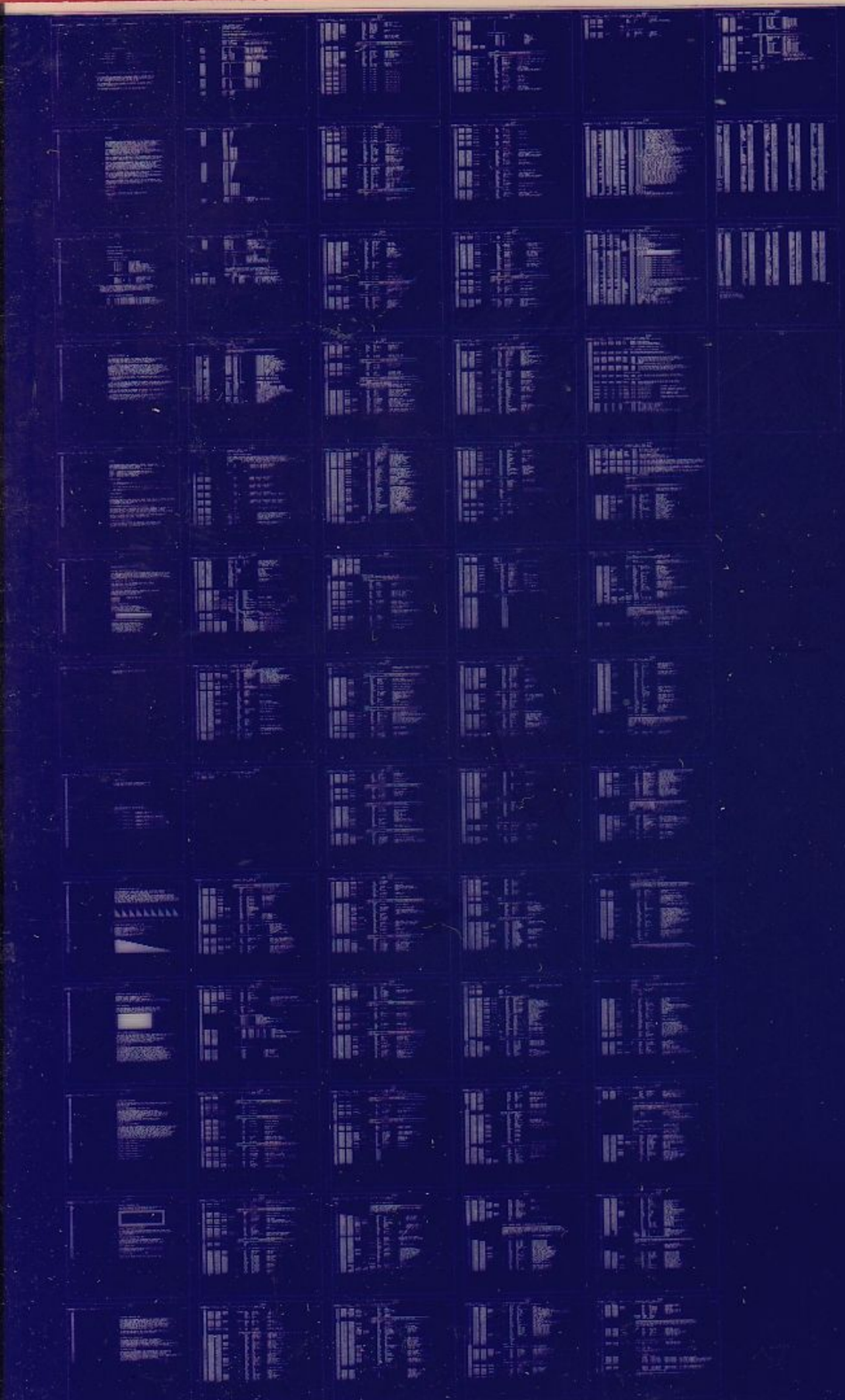
COPYRIGHT © 74-77

FICHE 1 OF 1

DEC 1977

digital

MADE IN USA



.REM *

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZVTC-E-D
PRODUCT NAME: VT50A, B, H, 52 ACCEPTANCE TEST
PROGRAM DATE: APRIL 1977

MAINTAINER: DIAGNOSTIC GROUP

COPYRIGHT (C) 1974,1977 DIGITAL EQUIPMENT CORP., MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DEC'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DEC.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

1. ABSTRACT

THIS PROGRAM IS AN ACCEPTANCE TEST OF THE VT50/52 VIDEO TERMINAL.
THE PROGRAM CONSISTS OF FOUR PARTS, ALL OF WHICH REQUIRE
OPERATOR INSPECTION OR INTERACTION. THE PROGRAM IS CAPABLE
OF HANDLING MULTIPLE UNITS IN A SEQUENTIAL DL-11 FASHION (REF 8.2).

****
ONLY ONE VT50/52 IS TESTED AT ONE TIME.
****

THE PROGRAM WILL DEFAULT TO THE CONSOLE TTY (REF 5.0 AND 8.2).
ALL CHARACTERS AND COMMANDS ARE TESTED. IN THE KEYBOARD CHARACTER
TEST THE FOLLOWING "FUNCTION" KEYS ARE NOT TESTED:
BREAK, REPEAT, AUTO-PRINT, AND SCROLL.

PART 1 CONSISTS OF A SERIES OF TEST PATTERNS DISPLAYED ON THE
VT50/52 SCREEN AND COPIER (REF 9, FOR DESCRIPTION). THE OPERATOR
MUST VISUALLY INSPECT EACH TEST PATTERN FOR ERROR DETECTION.

PART 2 IS A KEYBOARD CHARACTER TEST. THIS TEST IS TO DETERMINE
THAT THE TERMINAL IS GENERATING THE EXPECTED ASCII CODES.
IN THIS TEST AN OPERATOR WILL BE REQUIRED TO FOLLOW THE
INSTRUCTIONS DISPLAYED ON THE VT50/52 SCREEN AND EXECUTE THEM.
DUE TO THE FLEXIBILITY OF DIFFERENT PROCESSORS OR OPTIONS,
PARITY BIT TESTING MUST BE SELECTED BY THE OPERATOR. THE OPERATOR
SELECTS THE TYPE OF PARITY TO BE TESTED BY SW 00-01

PART 3 IS A KEYBOARD OCTAL VALUE LOOP. WHEN A KEY IS DEPRESSED,
THE OCTAL VALUE WILL BE DISPLAYED ON THE SCREEN.
IF THE KEY DEPRESSED WAS PRINTABLE, IT WILL ALSO BE DISPLAYED
ON THE SCREEN. IF A "DEFINED" CHARACTER, A TWO LETTER EQUALIVENT
(IE. BL=BELL, ES=ESCAPE ETC.) WILL BE DISPLAYED.

PART 4 IS A KEYBOARD ECHO LOOP. WHEN A KEY IS DEPRESSED, THE
CHARACTER IS ECHOED TO THE SCREEN. NO TESTING OF THE CHARACTER
IS PERFORMED. THIS ALLOWS THE OPERATOR A 'LOCAL' MODE OF OPERATION
BETWEEN THE VT50/52 AND THE HOST COMPUTER.

2. REQUIREMENTS

2.1 EQUIPTMENT

PDP-11 FAMILY COMPUTER WITH 8K WORDS OF MEMORY
VT50A, B, H, 52 VIDEO TERMINAL CONNECTED VIA A DL-11A/B TYPE
INTERFACE.

2.2 STORAGE

THIS PROGRAM USES 8K OF MEMORY.

3. LOADING PROCEDURE

PROCEDURE FOR NORMAL BINARY TAPES SHOULD BE FOLLOWED.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

STANDARD PDP-11 FORMAT

SW 15 = 1		HALT ON ERROR
SW 14 = 1		LOOP ON TEST
SW 13 = 1		INHIBIT ERROR TYPEOUTS
SW 12 = 1		INHIBIT PROGRAM SUB-TEST DELAY
SW 11 = 1		INHIBIT COPIER TESTING
SW 10 = 1		ENABLE "SAVE COPIER PAPER" MODE
SW 08 = 1		LOOP ON TEST IN SWR <4:0>
SW 07 = 1		KEYBOARD CONTROL OF THE TEST <SW 8 AND SW 7 = 1 IS AN ERROR>

KEYBOARD CHARACTER TEST ONLY

SW02 =	1	ENABLE PARITY BIT TEST
SW00-01=	00	EVEN PARITY CHECK
SW00-01=	01	ODD PARITY CHECK
SW00-01=	10	ALWAYS A 0
SW00-01=	11	ALWAYS A 1

SPECIAL NOTE: IF THE COMPUTER UTILIZED IS A LSI 11 OR A COMPUTER WITHOUT A SWITCH REGISTER, THE PROGRAM WILL UTILIZE LOCATIONS 174 AND 176 AS A "DISPLAY" REGISTER AND A "SWITCH" REGISTER RESPECTIVELY. THE OPERATOR WILL BE RESPONDSIBLE FOR THE LOADING OF THE "SWITCH" REGISTER LOCATION PRIOR TO STARTING OR RESTARTING THE PROGRAM.

4.2 STARTING ADDRESS OR ADDRESSES

200	IS THE STARTING ADDRESS OF THE ACCEPTANCE TEST
204	IS THE RESTART ADDRESS OF THE ACCEPTANCE TEST
210	IS THE STARTING ADDRESS OF THE KEYBOARD CHARACTER TEST
214	IS THE STARTING ADDRESS OF THE KEYBOARD OCTAL VALUE LOOP
220	IS THE STARTING ADDRESS OF THE KEYBOARD ECHO LOOP
224	IS THE SPECIAL STARTING ADDRESS FOR VT-50 PRODUCTION

5. OPERATING PROCEDURE

THE OPERATOR MUST INSERT THE CORRECT INFORMATION IN THE SWITCH REGISTER WHEN REQUIRED BY THE PROGRAM OR AN ERROR WILL OCCUR. ONCE STARTED, THE TEST WILL RUN IN ITS NORMAL MANNER WITHOUT OPERATOR INTERVENTION OR SWITCH CHANGES.

THIS PROGRAM ALLOWS THE OPERATOR TWO MODES OF TEST PATTERN SELECTION. THESE MODES ARE SELECTED BY THE STATE OF SW 07 AT THE BEGINNING OF THE PROGRAM. WHEN SW 07 IS A ZERO, THE PROGRAM IS UNDER SWITCH REGISTER CONTROL FOR TEST PATTERN SELECTION. IF SW07 IS EQUAL TO A ONE, THE PROGRAM IS UNDER KEYBOARD CONTROL OF THE TEST PATTERN SELECTION. IN THIS MODE THE OPERATOR WILL BE REQUIRED TO TYPE IN ON THE CONSOLE TTY THE FIRST AND LAST OCTAL BASE ADDRESS OF THE DL-11'S TO WHICH VT-50'S ARE CONNECTED.

IN THE KEYBOARD SELECT MODE, TWO CHARACTERS ARE USED TO SELECT THE "STARTING WITH" OR "LOOPING ON" A PATICULAR TEST PATTERN BY "/" OR "\" RESPECTFULY.

THE "/" KEY IS USED TO SUSPEND THE CURRENT TEST AND ASK THE OPERATOR A WHICH TEST PATTERN HE/SHE WISHES TO START. THE OPERATOR NOW DEPRESSES THE LETTER WHICH REPRESENTS THE TEST PATTERN TO BE STARTED WITH. REFER TO THE PROGRAM LISTING TABLE OF CONTENTS FOR THE TEST LETTER OF EACH PATTERN.

THE "\" KEY IS USED TO SUSPEND THE CURRENT TEST AND ASK THE OPERATOR WHICH TEST PATTERN HE/SHE WISHES TO LOOP ON. THE OPERATOR NOW DEPRESSES THE LETTER OF THE TEST TO LOOP ON.

IF DURING THE EXECUTION OF A TEST PATTERN, A KEY IS DEPRESSED AND SW 07 EQUALS A ZERO, AN ERROR WILL BE REPORTED TO THE CONSOLE TTY. IF SW 07 EQUALS A ONE, AND THE CHARACTER RECEIVED WAS NOT A "/" OR "\", AN ERROR WILL BE REPORTED. THE CODES "X-OFF" AND "X-ON" ARE THE ONLY EXCEPTIONS.

6. ERRORS

THIS PROGRAM USES THE DIAGNOSTIC 'SYSMAC' PACKAGE FOR
ERROR REPORTING AND TYPEOUT. REFER TO THE "ERROR POINTER TABLE"
FOR TYPE AND DESCRIPTION OF ERRORS.
THE ERROR INFORMATION CONSISTS OF THE FOLLOWING:

ERRPC - LOCATION AT WHICH AN ERROR WAS DETECTED
VTNOW - CURRENT DL-11 BUS ADDRESS OF VT50/52 UNDER TEST
TSTNUM- TEST PATTERN NUMBER OF FAILING TEST
EXPT - EXPECTED INPUT CHARACTER
RCVD - RECEIVED INPUT CHARACTER

7. RESTRICTIONS

A. THE OPERATOR SHOULD SET SW 15 AND 13 IF THE VT50/52 UNDER TEST
   IS THE CONSOLE TTY.

B. ONLY ONE VT50/52 CAN BE TESTED AT ONE TIME.

C. THE FIRST TIME AFTER LOADING THE PROGRAM, THE TERMINAL IDENTIFIER
   MUST BE RUN.

8. MISCELLANEOUS

8.1 EXECUTION TIME

EXECUTION TIME WILL VARY WITH THE "BAUD" RATE, AND IF A COPIER IS CONNEC
THE PROGRAM WILL TYPE 'END PASS' ON THE CONSOLE WHEN A PASS HAS BEEN COM
THE KEYBOARD LOOP AND CHARACTER TEST WILL NOT EXIT UNTIL
THE PROGRAM IS RESTARTED.

8.2 DEVICE ADDRESS PROGRAM LOCATIONS (AT APPROX. 1240)

THE LOCATION "FIRST" CONTAINS THE FIRST DL11 ADDRESS IF SEVERAL VT-50'S
ARE BEING TESTED. THE DEFAULT IS THE CONSOLE ADDRESS (177560)
THE LOCATION "LAST" CONTAINS THE LAST DL11 ADDRESS IF SEVERAL VT-50'S
ARE BEING TESTED. LOCATION VTNOW CONTAINS THE CURRENT DL11 BASE ADDRESS.

*NOTE: IF THESE LOCATIONS ARE CHANGED, THE OPERATOR MUST START
THE TEST AGAIN AT LOC. 200. THE PROGRAM WILL USE THE BASE
ADDRESS TO UPDATE THE ACTUAL PROGRAM VALUES.

8.3 COPIER SAVE PAPER SWITCH

IF SW 10 = 1 AND A COPIER IS INSTALLED, THE COPIER TESTS
WILL BE EXECUTED ON THE FIRST PASS AND THEN BYPASSED FOR
THE NEXT TEN PASSES. THIS REDUCES PAPER USAGE WHEN THE
PROGRAM IS RUN FOR AN EXTENDED PERIOD. (LOC. PTCT IS # OF PASSES)

9. PROGRAM DESCRIPTION <SCREEN>

9.1 A TERMINAL IDENTIFICATION TEST

THIS TEST WILL INTERROGATE THE VT50/52 UNDER TEST AS TO IT'S TYPE.
THE RESPONSE RECIEVED IS USED TO DETERMINE THE MODES TO BE TESTED
(IE. COPIER, 12/24 LINES, D.C.A <DIRECT CURSOR ADDRESSING> ETC.)
IF AN UNDEFINED OR AN INCORRECT RESPONSE IS RECIEVED, THE PROGRAM
ASSUMES VT50A MODEL (IE. 12 LINES, NO COPIER, NO D.C.A., NO EXTRA KEYPAD

9.2 B FULL SCREEN OF THE LETTER E

THIS TEST WILL FILL THE SCREEN WITH THE LETTER E.
THIS TEST WILL LOAD THE SCREEN RAM WITH A SINGLE
CHARACTER IN ALL LOCATIONS

9.3 C DATA PATH AND RAM NOISE TEST

THIS TEST WILL PRODUCE TWO FULL SCREENS OF A WORST CASE
NOISE PATTERN FOR THE SCREEN RAM AND THE DATA PATH.
ALTERNATING LINES OF THE FOLLOWING
PATTERNS SHOULD BE DISPLAYED.

```
*U*U*U*U*U*U        CODES 52 AND 125
ª'ª'ª'ª'ª'ª'        CODES 77 AND 100
*U*U*U*U*U*U
ª'ª'ª'ª'ª'ª'
```

9.4 D SIMPLE CHARACTER SET

ONE FULL LINE OF EACH CHARACTER
(CODES 40 THRU 137) OF THE CHARACTER
SET. THIS WILL ALSO TEST THAT ALL WORDS
IN THE SCREEN RAM CAN BE LOADED.
THIS WILL ALSO EXERCISE THE SCROLLING FUNCTION.

IE:
```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
DDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD
```

9.5 E INCREMENTING AND SLIDING CHARACTER SET

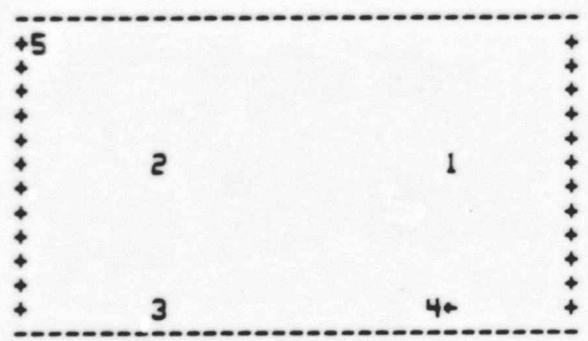
ONE FULL LINE OF AN INCREMENTING
CHARACTER SET ACROSS THE FULL SCREEN
STARTING WITH THE CODE 40 (SPACE)
THE NEXT LINE SHOULD BEGIN WITH THE
"!" CHARACTER (CODE 41) ETC. CONTINUE
THE PATTERN BY INCREMENTING THE
FIRST COLUMN CHARACTER UNTIL THE
FULL CHARACTER SET HAS BEEN EXHAUSTED.

```
;311
;
;
;
```

9.6 F CURSOR MOTION

IN THIS TEST THE BASIC CURSOR MOTIONS ARE TESTED. THE FOLLOWING TEST PATTERN IS GENERATED TO TEST THE CURSOR FUNCTIONS

BEFORE



UPON COMPLETION OF THE SETUP THE BLINKING CURSOR WILL BE TO THE RIGHT OF #4.

TO TEST "CURSOR UP": GENERATE CURSOR UP 6/12 TIMES AND DISPLAY A "X"

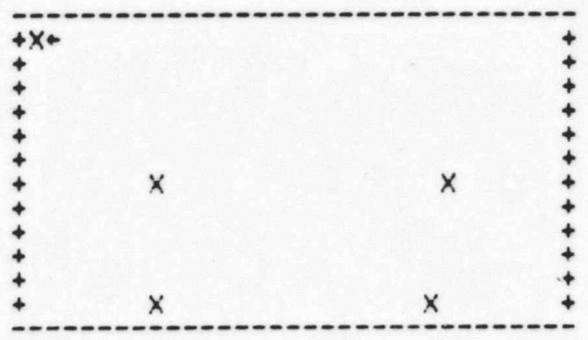
TO TEST "CURSOR LEFT": GENERATE CURSOR LEFT (BACKSPACE) FOURTY FOUR TIMES AND DISPLAY A "X"

TO TEST "CURSOR DOWN": GENERATE CURSOR DOWN 6/12 TIMES AND DISPLAY A "X"

TO TEST "CURSOR RIGHT": GENERATE CURSOR RIGHT FOURTY TWO TIMES AND DISPLAY A "X"

TO TEST "CURSOR HOME": GENERATE CURSOR HOME AND DISPLAY A "X"

AFTER



9.7 G TAB, BACKSPACE AND BELL TEST

THIS PATTERN WILL TEST THAT THE 'TAB' FUNCTIONS CORRECTLY FROM EACH
COLUMN POSITION. GENERATE A REFERENCE LINE USING SPACES
AND THE LETTER 'I'. ON THE NEXT LINE
RING THE BELL AND DISPLAY ONE CHARACTER IN COLUMN ONE AND EXECUTE A 'TAB
THE CURSOR SHOULD THEN BE POSITIONED BELOW THE REFERENCE TAB STOP.
REPEAT THE ONE CHARACTER IN THE FIRST TAB COLUMN ACROSS THE SCREEN.
ON THE NEXT LINES USE INCREASING NUMBERS OF CHARACTERS BEFORE THE 'TAB'
THE BELL WILL RING AT THE BEGINNING OF EACH LINE. THE SECOND SECTION TES
THE BACKSPACE CODE. A FULL LINE OF THE CHARACTER "X" WILL BE DISPLAYED
STARTING AT THE RIGHT EDGE AND EXTENDING TO THE LEFT.

```
--------------------------------------------------------------------------------
+I      I       I       I       I       I       I       I       I
+I      I       I       I       I       I       I       I       I
+IA     IA      IA      IA      IA      IA      IA      IA      IA
+IAA    IAA     IAA     IAA     IAA     IAA     IAA     IAA     IAA
+IAAA   IAAA    IAAA    IAAA    IAAA    IAAA    IAAA    IAAA    IAAA
+IAAAA  IAAAA   IAAAA   IAAAA   IAAAA   IAAAA   IAAAA   IAAAA   IAAAA
+IAAAAA IAAAAA  IAAAAA  IAAAAA  IAAAAA  IAAAAA  IAAAAA  IAAAAA  IAAAAA
+IAAAAAA IAAAAAA IAAAAAA IAAAAAA IAAAAAA IAAAAAA IAAAAAA IAAAAAA IAAAAAA

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

9.10 H ERASE FROM CURSOR TO END OF LINE:

H.1 GENERATE A FULL SCREEN OF THE CHARACTER "E"
H.2 EXECUTE A "CURSOR HOME"
H.3 EXECUTE "ERASE FROM CURSOR TO END OF LINE"
H.4 EXECUTE "CURSOR RIGHT" 6/3 TIMES
H.5 EXECUTE "LF"
REPEAT H.3 THRU H.5 12/24 TIMES
THE END DISPLAY SHOULD APPEAR TO BE A
STAIR STEP PATTERN

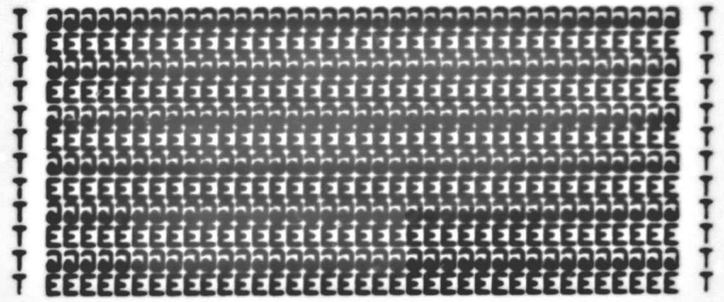
```
EEEEEE
EEEEEEEEEEEE
EEEEEEEEEEEEEEEEEE
EEEEEEEEEEEEEEEEEEEEEE
EEEEEEEEEEEEEEEEEEEEEEEEEEEE
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
```

9.11 I ERASE FROM CURSOR TO END OF THE SCREEN:

GENERATE A FULL SCREEN OF THE CHARACTER "E"
EXECUTE FOURTY "CURSORS LEFT"
EXECUTE "ERASE SCREEN" AND "CURSOR UP"
REPEAT THIS 12 OR 24 TIMES, UPON COMPLETION THE PATTERN WILL
BE A LINE OF FOURTY CHARACTERS AT THE TOP LEFT OF THE SCREEN.

9.12 J VIDEO COUPLING:

THIS PATTERN WILL ALLOW FOR THE CHECKING OF VIDEO
COUPLING BETWEEN THE VIDEO AND VERTICAL CIRCUITS.
IF COUPLING OCCURS, THE VERTICAL DOTS OF THE CHARACTERS
WILL BE UNEVENLY SPACED OR DOTS WILL SPARKLE.



9.13 K DIRECT CURSOR ADDRESSING (D.C.A.)

THIS TEST IS ONLY EXECUTED ON A VT50H. WHEN EXECUTED ON A VT50H
THIS TEST WILL BE RUN TWO TIMES. THE FIRST TIME WILL BE WITH
AN "ESC-Y" AND THE SECOND WITH AN "CODE 16".
IF AN INCORRECT I.D., THIS TEST WILL NOT BE EXECUTED.
THIS TEST WILL RANDOMLY FILL THE SCREEN. THE END RESULT
WILL BE THE SAME STATEMENT ON ALL VERTICAL LINES.
EACH OF THE LINES SHOULD APPEAR THE SAME.

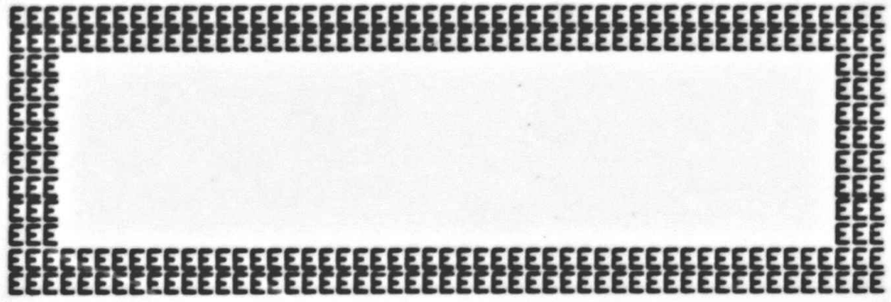
9.14 L HOLD SCREEN TEST

NOT EXECUTED IF VT50/52 PRODUCTION STARTING ADDRESS.
THIS TEST DETERMINES THAT THE "XON-XOFF" FEATURE IS FUNCTIONING.
A FULL SCREEN SCROLL IS EXECUTED AND A SUB-TEST TITLE
WILL BE DISPLAYED. THE HOLD SCREEN MODE IS ENABLED AND
THE PROGRAM WILL ATTEMP TO SCROLL THE SCREEN AGAIN AND SEND A MESSAGE.
THIS WILL CAUSE AN "X-OFF" TO BE SENT AND THE SCREEN IS
INHIBITED FROM SCROLLING. ANY ADDITIONAL CHARACTERS RECEIVED
WILL BE PLACED INTO THE SILO STORAGE BUFFER. UPON RECEIPT
OF AN "X-OFF", THE PROGRAM WILL DISABLE HOLD SCREEN MODE AND
SHOULD RECEIVE AN "X-ON". TO CHECK PROPER SILO OPERATION,
THE MESSAGE "TESTING SILO REG" SHOULD APPEAR UNDER THE SUB-TEST TITLE
FOLLOWED BY "SILO TEST DONE" ON THE NEXT LINE.

```
                        COPIER TEST PATTERNS

                        THE TEST PATTERNS USED TO TEST THE COPIER ARE THE SAME PATTERNS
                        AS SOME OF THE SCREEN TESTS. TWO ADDITIONAL PATTERNS HAVE
                        BEEN INCLUDED.

9.15 M                  GRAPHICS MODE/REVERSE LINE FEED TEST

                        THIS TEST IS EXECUTED ONLY IF UNIT IS A VT52.
                        GRAPHICS MODE IS ENTERED AND 37 LINES OF GRAPHICS
                        CHARACTERS(LINE LENGTH IS DECREMENTED BY ONE CHAR. AFTER
                        A LINE IS PRINTED)ARE GENERATED. THE BOTTOM LINE OF THE
                        WILL FORM A COMPLETE SERIES OF GRAPHIC CHARACTERS WHICH
                        CAN BE VERIFIED FOR ACCURACY.
                        IF REVERSE LINE FEED IS NOT OPERATIONAL A SINGLE
                        LINE OF GRAPHICS CHARACTERS WILL BE GENERATED
                        ON LINE 0 ONLY.

9.16 N                  COPIER - AUTO COPY MODE

                        THIS TEST IS USED TO DETERMINE IF THE AUTO COPY MODE IS FUNCTIONING CORR
                        THE SUB-TEST TITLE AND A ROW OF THE LETTER 'E' WILL BE DISPLAYED.
                        THE AUTO-COPY MODE IS ENABLED. THE COPIER SHOULD THEN START.
                        A PROGRAM DELAY IS EXECUTED BETWEEN EACH LINE. THE RESULT
                        IS THE COPIER SHOULD COPY ONE LINE AND THEN STOP FOR THE PROGRAM
                        DELAY TIME. WHEN THE PROGRAM DELAY IS COMPLETED ANOTHER LINE
                        OF 'E'S WILL BE DISPLAYED AND THE COPIER SHOULD START AGAIN.
                        UPON COMPLETION OF THE 12 OR 24 LINES, DISABLE AUTO-COPY MODE
                        IS SENT TO THE VT50. IF THE AUTO COPY MODE IS NOT DISABLED
                        THE NEXT SUB-TEST WOULD PERFORM IN A SIMILAR MANNER.

9.17 O                  COPIER - FULL SCREEN OF THE LETTER E

                        SAME AS SCREEN PATTERN

9.20 P                  COPIER - DATA PATH TEST PATTERN

                        SAME AS SCREEN PATTERN

9.21 Q                  COPIER - SINGLE CHARACTER PER LINE

                        SAME AS SCREEN PATTERN

9.22 R                  COPIER - ROTATING CHARACTER TEST

                        SAME AS SCREEN PATTERN
```

9.23 S COPIER - PERIMETER TEST

THIS TEST COPIES THE PERIMETER OF THE SCREEN AND THE
RESULTING PICTURE SHOULD RESEMBLE BELOW



9.24 T COPIER - DISCLAIMER STATEMENT

IN THIS TEST THE DISCLAMER STATEMENT FROM THE COVER PAGE
OF THIS DOCUMENT IS DISPLAYED ON THE SCREEN AND THEN COPIED.
THE COPIED VERSION SHOULD BE COMPARED TO THE FRONT COVER TO
CHECK FOR ERRORS.

9.25 U PRINTER CONTROLLER MODE TEST

A 'ROLLING' PATTERN OF INCREMENTING CHARACTERS IS
ISSUED TO THE UNIT AFTER PRINTER CONTROLLER
MODE HAS BEEN ENTERED. 92 LINES(OF 132 CHAR. EACH)
SHOULD BE PRINTED WITH NO DATA APPERING ON SCREEN.

9.26 V PRINT SCREEN TEST

THE SCREEN IS FILLED WITH 24 LINES OF 'E'S. THE
PRINT SCREEN ESCAPE SEQUENCE IS THE ISSUED AND ALL
24 LINES SHOULD BE PRINTED.

9.27 W AUTO PRINT TEST

OPERATION IS SAME AS AUTO COPY TEST EXCEPT THAT
THE PRINTER,RATHER THAN THE COPIER IS THE DESTINATION.

```
579
580
581
582
583                                     9.30 X  KEYBOARD CHARACTER TEST
584
585                                             THIS TEST IS DESIGNED TO VERIFY THAT CORRECT CHARACTER CODES
586                                             AND PARITY BIT ARE GENERATED WHEN A KEY IS DEPRESSED.
587                                             THIS TEST REQUIRES THE OPERATOR TO EXECUTE THE
588                                             INSTRUCTIONS DISPLAYED ON THE SCREEN. THE OPERATOR SHOULD
589                                             ONLY DEPRESS ONE KEY AT A TIME, WITH SOME EXCEPTIONS.
590                                             THE OPERATOR WILL BE REQUIRED TO SKIP THOSE KEYS THAT
591                                             ARE NOT IMPLEMENTED IF THE UNIT IS A VT50. THE PROGRAM WILL
592                                             INFORM THE OPERATOR WHICH ROW TO TEST.
593
594                                             IN TESTING THE PARITY BIT, SW 0 AND 1 ARE USED TO
595                                             INFORM THE PROGRAM OF THE EXPECTED PARITY. AN INCORRECT
596                                             SWITCH SETTING WILL RESULT IN AN ERROR.
597
598
599                                     9.31 Y  KEYBOARD OCTAL VALUE LOOP
600
601                                             THIS LOOP IS PROVIDED TO ENABLE THE OPERATOR
602                                             TO EXAMINE THE OCTAL VALUE OF A CHARACTER. WHEN A KEY IS
603                                             DEPRESSED, THE OCTAL VALUE WILL BE DISPLAYED.
604                                             IF THE CHARACTER WAS A PRINTABLE CHARACTER, IT WILL BE DISPLAYED.
605                                             THOSE CODES DEFINED AS "CONTROL" WILL BE DISPLAYED AS A
606                                             TWO LETTER MNEMONIC (IE. DE=DELETE, BL=BELL, CL=CURSOR LEFT ETC.)
607
608                                     9.32 Z  KEYBOARD ECHO LOOP
609
610                                             WHEN A KEY IS DEPRESSED, THE CHARACTER WILL BE DISPLAYED.
611                                             NO MODIFICATION OR DATA TEST IS PERFORMED. THIS TEST CAN
612                                             BE USED TO DETERMINE IF THERE IS A "UART" OR SERIAL
613                                             LINE PROBLEM. WHEN THE VT50/52 IS IN LOCAL MODE (NO UART/SERIAL LINE)
614                                             THE CHARACTERS SHOULD BE ECHOED CORRECTLY. IF NOT, THE PROBLEM
615                                             IS IN THE VT50 UNIT. IF CHANGED TO REMOTE
616                                             MODE AND THE CHARACTERS ARE IN ERROR, THERE IS A UART/
617                                             SERIAL LINE PROBLEM.
618
```

```
630                             .TITLE  MAINDEC-11-DZVTC-E
(1)                             ;*COPYRIGHT (C) 1977
(1)                             ;*DIGITAL EQUIPMENT CORP.
(1)                             ;*MAYNARD, MASS. 01754
(1)                             ;*
(1)                             ;*PROGRAM BY TERMINAL DIAGNOSTICS
(1)                             ;*
(1)                             ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
(1)                             ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
(1)                             ;*
631                             .SBTTL  BASIC DEFINITIONS
(1)
(1)                             ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
(1)             001100          STACK=  1100
(1)                             .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
(1)                             .EQUIV  TOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL
(1)
(1)                             ;*MISCELLANEOUS DEFINITIONS
(1)             000011          HT=     11              ;;CODE FOR HORIZONTAL TAB
(1)             000012          LF=     12              ;;CODE FOR LINE FEED
(1)             000015          CR=     15              ;;CODE FOR CARRIAGE RETURN
(1)             000200          CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
(1)             177776          PS=     177776          ;;PROCESSOR STATUS WORD
(1)                             .EQUIV  PS,PSW
(1)             177774          STKLMT= 177774          ;;STACK LIMIT REGISTER
(1)             177772          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
(1)             177570          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
(1)             177570          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
(1)
(1)                             ;*GENERAL PURPOSE REGISTER DEFINITIONS
(1)             000000          R0=     %0              ;;GENERAL REGISTER
(1)             000001          R1=     %1              ;;GENERAL REGISTER
(1)             000002          R2=     %2              ;;GENERAL REGISTER
(1)             000003          R3=     %3              ;;GENERAL REGISTER
(1)             000004          R4=     %4              ;;GENERAL REGISTER
(1)             000005          R5=     %5              ;;GENERAL REGISTER
(1)             000006          R6=     %6              ;;GENERAL REGISTER
(1)             000007          R7=     %7              ;;GENERAL REGISTER
(1)             000006          SP=     %6              ;;STACK POINTER
(1)             000007          PC=     %7              ;;PROGRAM COUNTER
(1)
(1)                             ;*PRIORITY LEVEL DEFINITIONS
(1)             000000          PR0=    0               ;;PRIORITY LEVEL 0
(1)             000040          PR1=    40              ;;PRIORITY LEVEL 1
(1)             000100          PR2=    100             ;;PRIORITY LEVEL 2
(1)             000140          PR3=    140             ;;PRIORITY LEVEL 3
(1)             000200          PR4=    200             ;;PRIORITY LEVEL 4
(1)             000240          PR5=    240             ;;PRIORITY LEVEL 5
(1)             000300          PR6=    300             ;;PRIORITY LEVEL 6
(1)             000340          PR7=    340             ;;PRIORITY LEVEL 7
(1)
(1)                             ;*"SWITCH REGISTER" SWITCH DEFINITIONS
(1)             100000          SW15=   100000
(1)             040000          SW14=   40000
(1)             020000          SW13=   20000
(1)             010000          SW12=   10000
```

```
(1)  004000                  SW11=   4000
(1)  002000                  SW10=   2000
(1)  001000                  SW09=   1000
(1)  000400                  SW08=   400
(1)  000200                  SW07=   200
(1)  000100                  SW06=   100
(1)  000040                  SW05=   40
(1)  000020                  SW04=   20
(1)  000010                  SW03=   10
(1)  000004                  SW02=   4
(1)  000002                  SW01=   2
(1)  000001                  SW00=   1
(1)                          .EQUIV  SW09,SW9
(1)                          .EQUIV  SW08,SW8
(1)                          .EQUIV  SW07,SW7
(1)                          .EQUIV  SW06,SW6
(1)                          .EQUIV  SW05,SW5
(1)                          .EQUIV  SW04,SW4
(1)                          .EQUIV  SW03,SW3
(1)                          .EQUIV  SW02,SW2
(1)                          .EQUIV  SW01,SW1
(1)                          .EQUIV  SW00,SW0
(1)
(1)                          ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(1)  100000                  BIT15=  100000
(1)  040000                  BIT14=  40000
(1)  020000                  BIT13=  20000
(1)  010000                  BIT12=  10000
(1)  004000                  BIT11=  4000
(1)  002000                  BIT10=  2000
(1)  001000                  BIT09=  1000
(1)  000400                  BIT08=  400
(1)  000200                  BIT07=  200
(1)  000100                  BIT06=  100
(1)  000040                  BIT05=  40
(1)  000020                  BIT04=  20
(1)  000010                  BIT03=  10
(1)  000004                  BIT02=  4
(1)  000002                  BIT01=  2
(1)  000001                  BIT00=  1
(1)                          .EQUIV  BIT09,BIT9
(1)                          .EQUIV  BIT08,BIT8
(1)                          .EQUIV  BIT07,BIT7
(1)                          .EQUIV  BIT06,BIT6
(1)                          .EQUIV  BIT05,BIT5
(1)                          .EQUIV  BIT04,BIT4
(1)                          .EQUIV  BIT03,BIT3
(1)                          .EQUIV  BIT02,BIT2
(1)                          .EQUIV  BIT01,BIT1
(1)                          .EQUIV  BIT00,BIT0
(1)
(1)                          ;*BASIC "CPU" TRAP VECTOR ADDRESSES
(1)  000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
(1)  000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
(1)  000014                  TBITVEC=14              ;;"T" BIT
(1)  000014                  TRTVEC= 14              ;;TRACE TRAP
```

```
                    000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
                    000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                    000024                  PWRVEC= 24              ;;POWER FAIL
                    000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
                    000034                  TRAPVEC=34              ;;"TRAP" TRAP
                    000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
                    000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
                    000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
                                            .SBTTL  OPERATIONAL SWITCH SETTINGS
                                            ;*
                                            ;*      SWITCH                  USE
                                            ;*      ------                  ---
                                            ;*        15                    HALT ON ERROR
                                            ;*        14                    LOOP ON TEST
                                            ;*        13                    INHIBIT ERROR TYPEOUTS
                                            ;*        12                    INHIBIT SUB-TEST DELAY'S
                                            ;*        10                    ENABLE SAVE COPIER PAPER MODE
                                            ;*         8                    LOOP ON TEST IN SWR<7:0>
                                            .SBTTL  TRAP CATCHER
                    000000                          .=0
                                            ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ",.+2,HALT"
                                            ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                            ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                    000174                          .=174
000174  000000                 DISPREG:        .WORD   0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                 SWREG:          .WORD   0               ;;SOFTWARE SWITCH REGISTER
                                            .SBTTL  STARTING ADDRESS(ES)
000200  000137  001334                  JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137  001364                  JMP     RBEGIN          ;JUMP TO RESTART ADDRESS
000210  000137  001410                  JMP     BEGIN1          ;JUMP TO KEYBOARD CHARACTER TEST
000214  000137  001420                  JMP     BEGIN2          ;JUMP TO CHAR OCTAL VALUE LOOP
000220  000137  001400                  JMP     BEGIN3          ;JUMP TO ASCII ECHO LOOP
000224  000137  001360                  JMP     MANFU           ;JUMP TO W.F. SPECIAL TEST PARAM.
```

```
                        .SBTTL  COMMON TAGS

                        ;;**********************************************************************
                        ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                        ;*USED IN THE PROGRAM.

                001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000  $PASS:  .WORD   0                       ;;CONTAINS PASS COUNT
001102  000     $TSTNM: .BYTE   0                       ;;CONTAINS THE TEST NUMBER
001103  000     $ERFLG: .BYTE   0                       ;;CONTAINS ERROR FLAG
001104  000000  $ICNT:  .WORD   0                       ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000  $LPADR: .WORD   0                       ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000  $LPERR: .WORD   0                       ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000  $ERTTL: .WORD   0                       ;;CONTAINS TOTAL ERRORS DETECTED
001114  000     $ITEMB: .BYTE   0                       ;;CONTAINS ITEM CONTROL BYTE
001115  001     $ERMAX: .BYTE   1                       ;;CONTAINS MAX. ERRORS PER TEST
001116  000000  $ERRPC: .WORD   0                       ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000  $GDADR: .WORD   0                       ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000  $BDADR: .WORD   0                       ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000  $GDDAT: .WORD   0                       ;;CONTAINS 'GOOD' DATA
001126  000000  $BDDAT: .WORD   0                       ;;CONTAINS 'BAD' DATA
001130  000000          .WORD   0                       ;;RESERVED--NOT TO BE USED
001132  000000          .WORD   0
001134  000     $AUTOB: .BYTE   0                       ;;AUTOMATIC MODE INDICATOR
001135  000     $INTAG: .BYTE   0                       ;;INTERRUPT MODE INDICATOR
001136  000000          .WORD   0
001140  177570  $SWR:    .WORD   DSWR                  ;;ADDRESS OF SWITCH REGISTER
001142  177570  DISPLAY: .WORD   DDISP                 ;;ADDRESS OF DISPLAY REGISTER
001144  177560  $TKS:    177560                        ;;TTY KBD STATUS
001146  177562  $TKB:    177562                        ;;TTY KBD BUFFER
001150  177564  $TPS:    177564                        ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566  $TPB:    177566                        ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000     $NULL:  .BYTE   0                       ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002     $FILLS: .BYTE   2                       ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012     $FILLC: .BYTE   12                      ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157  000     $TPFLG: .BYTE   0                       ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000  $REGAD: .WORD   0                       ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH  ($REG0) WAS OBTAINED
001162  000000  $REG0:  .WORD   0                       ;;CONTAINS (($REGAD)+0)
001164  000000  $REG1:  .WORD   0                       ;;CONTAINS (($REGAD)+2)
001166  077     $QUES:  .ASCII  /?/                     ;;QUESTION MARK
001167  015     $CRLF:  .ASCII  <15>                    ;;CARRIAGE RETURN
001170  000012  $LF:    .ASCIZ  <12>                    ;;LINE FEED
                        ;;**********************************************************************
```

```
                                .SBTTL  ERROR POINTER TABLE

                                ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                ;*NOTE2:      EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                                ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                                ;*      DH              ;;POINTS TO THE DATA HEADER
                                ;*      DT              ;;POINTS TO THE DATA
                                ;*      DF              ;;POINTS TO THE DATA FORMAT

(1)  001172                     $ERRTB:

641
643
644                             ;ITEM   1
645  001172  020345                     EM1             ;ERROR FLAG ON HOST TRANSMIT STATUS
646  001174  020570                     DH1             ;ERRPC  VTNOW   TSTNUM
647  001176  020734                     DT1             ;$ERRPC VTNOW   TSTNUM
648  001200  000000                     0
649
650                             ;ITEM   2
651  001202  020412                     EM2             ;NO HOST INPUT FLAG RECEIVED
652  001204  020570                     DH1             ;ERRPC  VTNOW   TSTNUM
653  001206  020734                     DT1             ;$ERRPC VTNOW   TSTNUM
654  001210  000000                     0
655
656                             ;ITEM   3
657  001212  020441                     EM3             ;INCORRECT I.D. RESPONSE
658  001214  020617                     DH3             ;ERRPC  VTNOW   1ST WD  2ND WD  3RD WD
659  001216  020744                     DT3             ;$ERRPC VTNOW   SAVE4   SAVE2   SAVE3
660  001220  000000                     0
661
662                             ;ITEM   4
663  001222  020465                     EM4             ;INCORRECT OR UNEXPECTED INPUT CHARACTER
664  001224  020666                     DH4             ;ERRPC  VTNOW   TSTNUM  EXPT    RCVD
665  001226  020760                     DT4             ;$ERRPC VTNOW   TSTNUM  $GDDAT $BDDAT
666  001230  000000                     0
667
668                             ;ITEM   5
669  001232  020530                     EM5             ;INVALID BUSS ADDRESS, TRY AGAIN
670  001234  000000                     0
671  001236  000000                     0
672  001240  000000                     0
673  001242  177560             FIRST:  177560                  ;FIRST DEVICE ADDRESS OF SEQUIENTAL DL-11-A/B TYPE DEVIC
674                                                             ;DEFAULT TO THE CONSOLE ADDRESS
675  001244  000000             LAST:   0                       ;LAST DEVICE ADDRESS OF DL-11-A/B TYPE
676  001246  177560             VTNOW:  177560                  ;CURRENT DEVICE BUSS ADDRESS
677  001250  100011             PTCT:   100011                  ;COPIER PAPER SAVE COUNT (LOW BYTE)
678  001252  000000             TSTNUM: 0                       ;ERROR PATTERN
679
680  001254  000300             TIMEO:  300                     ;CHARACTER FLAG TIMEOUT CONSTANT
681  001256  000005             SUBTST: 5                       ;SUBTEST DELAY CONSTANT
687  001260  000000             VTSXX:  0                       ;I.D. AND CHARASTICS
```

```
                                LASTLN: 53          ;OR 67          ;LAST VALID LINE # +40
                                TOTALC: 960.        ;OR 1920.       ;TOTAL CHARACTER COUNT
                                VHO:    12.         ;24.            ;VERTICAL LINE COUNT
                                VH1:    6.          ;12.            ;1/2 VERTICAL LINE COUNT
                                VH2:    3.          ;6.             ;1/4 VERTICAL LINE COUNT
                                PRTCNT: 0
                                VTIS:   177560                      ;DEVICE ADDRESSES
                                VTIB:   177562                      ;IN DATA
                                VTOS:   177564                      ;OUT STAT
                                VTOB:   177566                      ;OUT DATA
                                STCHAR: 0                           ;TEMP REG'S
                                LASTCH: 140         ;OR 200         ;FIRST NON-VALID CHARACTER
                                TEMP:   0                           ;TEMP REG'S
                                TEMP0:  0                           ;
                                PNTWID: 132.                        ;COLUMN COUNT FOR LINE PRINTER.
                                WIDTH:  80.                         ;COLUMN WIDTH
                                SAVE1:  0                           ;TEMP REG'S
                                SAVE2:  0
                                SAVE3:  0
                                SAVE4:  0
                                WFTEST: 0                           ;NON-ZERO IF SA = 224

                        BEGIN:  MOV     #TST2,WHERE             ;STARTING ACCEPTANCE TEST ADDRESS
                                CLR     SAVE4
                                CLR     WFTEST
                                CLR     PRTCNT
                                BR      GINA
                        MANFU:  INC     WFTEST                  ;SET W.F. PARAM.
                        RBEGIN: CLR     PRTCNT                  ;RESTART ADDRESS
                                MOV     #TST2,WHERE
                                BR      GIN
                        BEGIN3: MOV     #KRBECH,WHERE           ;START AT ECHO LOOP
                                BR      GIN
                        BEGIN1: MOV     #KRBTST,WHERE           ;STARTING CHARACTER TEST ADDRESS
                                BR      GIN
                        BEGIN2: MOV     #KRBECO,WHERE           ;STARTING CHARACTER LOOP ADDRESS
                        GIN:    MOV     #1,SAVE4
                        GINA:   RESET
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
                                MOV     #$CMTAG,R6              ;;FIRST LOCATION TO BE CLEARED
                                CLR     (R6)+                   ;;CLEAR MEMORY LOCATION
                                CMP     #SWR,R6 ;;DONE?
                                BNE     .-6                     ;;LOOP BACK IF NO
                                MOV     #STACK,SP               ;;SETUP THE STACK POINTER
                        ;;INITIALIZE A FEW VECTORS
                                MOV     #$SCOPE,@#IOTVEC        ;;IOT VECTOR FOR SCOPE ROUTINE
                                MOV     #340,@#IOTVEC+2         ;;LEVEL 7
                                MOV     #$ERROR,@#EMTVEC        ;;EMT VECTOR FOR ERROR ROUTINE
                                MOV     #340,@#EMTVEC+2         ;;LEVEL 7
                                MOV     #$TRAP,@#TRAPVEC        ;;TRAP VECTOR FOR TRAP CALLS
                                MOV     #340,@#TRAPVEC+2        ;LEVEL 7
                                MOV     #$PWRDN,@#PWRVEC        ;;POWER FAILURE VECTOR
                                MOV     #340,@#PWRVEC+2         ;;LEVEL 7
                                MOV     $ENDCT,$EOPCT           ;;SETUP END-OF-PROGRAM COUNTER
```

```
    (1)  001544  012737  001544  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
    (2)                                    ;;SIZE FOR A HARDWARE SWITCH REGISTER.  IF NOT FOUND OR IT IS
    (2)                                    ;;EQUAL TO A "-1"  SETUP FOR A SOFTWARE SWITCH REGISTER.
    (2)  001552  013746  000004                    MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
    (2)  001556  012737  001612  000004            MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
    (2)  001564  012737  177570  001140            MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
    (2)  001572  012737  177570  001142            MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
    (2)  001600  022777  177777  177332            CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
    (2)  001606  001012                            BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
    (2)                                                                    ;;AND  THE HARDWARE SWR IS NOT = -1
    (2)  001610  000403                            BR      65$             ;;BRANCH IF NO TIMEOUT
    (2)  001612  012716  001620            64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
    (2)  001616  000002                            RTI
    (2)  001620  012737  000176  001140    65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
    (2)  001626  012737  000174  001142            MOV     #DISPREG,DISPLAY
    (2)  001634  012637  000004            66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
    (1)
  728  001640  005037  010666                    CLR     IGNORE
  729  001644  005037  010650                    CLR     LOOP
  730  001650  012737  023610  000020            MOV     #MSCOPE,@#IOTVEC
  731  001656  005737  001330                    TST     SAVE4           ;TEST FLAG
  732  001662  001051                            BNE     SBEGIN          ;BR IF NON-ZERO
  733  001664  104401                            TYPE
  734  001666  013144                            TITLE
  735  001670  105777  177244                    TSTB    @SWR
  736  001674  100044                            BPL     SBEGIN          ;BR IF CLEARED
  737  001676  012737  001776  000004            MOV     #12$,4          ;SET UP SLAVE TIMEOUT
  738  001704  012737  000340  000006            MOV     #340,6          ;LOCK OUT ANY INTERRUPTS
  739  001712  104401                    11$:    TYPE
  740  001714  020156                            WHATO                   ;FIND OUT THE DEVICE ADDRESS
  741  001716  104410                            RDOCT
  742  001720  012637  001242                    MOV     (SP)+,FIRST     ;SAVE THE ADDRESS
  743  001724  022737  160000  001242            CMP     #160000,FIRST
  744  001732  101367                            BHI     11$             ;BR IF INVALID
  745  001734  005777  177302                    TST     @FIRST          ;TEST IF VALID
  746  001740  005037  001244                    CLR     LAST
  747  001744  104401                            TYPE
  748  001746  020221                            WHAT1                   ;FIND OUT THE LAST ADDRESS
  749  001750  104410                            RDOCT
  750  001752  012637  001244                    MOV     (SP)+,LAST      ;SAVE LAST ADDRESS
  751  001756  005777  177262                    TST     @LAST           ;TEST IF VALID
  752  001762  012737  000006  000004            MOV     #6,@#4
  753  001770  005037  000006                    CLR     @#6
  754  001774  000404                            BR      SBEGIN
  755  001776  022626                    12$:    CMP     (SP)+,(SP)+     ;ADJUST STACK FROM TIMEOUT
  756  002000  104401                            TYPE
  757  002002  016134                            INVLID                  ;INVALID ADDRESS MESSAGE
  758  002004  000742                            BR      11$             ;LET OPERATOR HAVE ANO TRY
  759  002006  013737  001242  001246    SBEGIN: MOV     FIRST,VTNOW     ;LOAD INITIAL DEVICE ADDRESS
  760
  761  002014  012700  001276            PSTRT:  MOV     #VTIS,R0        ;LOAD POINTER
  762  002020  013701  001246                    MOV     VTNOW,R1        ;LOAD INPUT STAT
  763  002024  010120                            MOV     R1,(R0)+
  764  002026  005721                            TST     (R1)+
  765  002030  010120                            MOV     R1,(R0)+
  766  002032  005721                            TST     (R1)+
```

```
   767  002034  010120                    MOV     R1,(R0)+
   768  002036  005721                    TST     (R1)+
   769  002040  010110                    MOV     R1,(R0)
```

```
                        ;;*********************************************************************
                        ;*TEST 1          A       TERMINAL IDENTIFICATION TEST
                        ;;*********************************************************************
002042  000004          TST1:   SCOPE
002044  004537  011442          JSR     R5,AMSG         ;DISPLAY HEADER
002050  013656          M914
002052  004537  011442          JSR     R5,AMSG         ;SEND REQUEST FOR IDENTIFICATION
002056  017551          RFI

002060  004737  013052          JSR     PC,GETCHR       ;GET A CHARACTER
002064  000537                  BR      2$              ;;BR BACK IF NO INPUT
002066  010037  001330          MOV     R0,SAVE4        ;SAVE RESPONSE
002072  004737  013052          JSR     PC,GETCHR       ;GET A CHAR.
002076  000532                  BR      2$              ;;BR IF NO INPUT
002100  010037  001324          MOV     R0,SAVE2
002104  004737  013052          JSR     PC,GETCHR       ;GET A CHAR.
002110  000525                  BR      2$              ;;BR IF NO INPUT
002112  010037  001326          MOV     R0,SAVE3
002116  042737  177600  001330  BIC     #177600,SAVE4   ;MASK BIT 7
002124  042737  177600  001324  BIC     #177600,SAVE2
002132  042737  177600  001326  BIC     #177600,SAVE3
002140  005737  001332          TST     WFTEST
002144  001402                  BEQ     10$
002146  004737  011370          JSR     PC,ADELAY       ;EXTRA DELAY
002152  122737  000033  001330  10$:    CMPB    #33,SAVE4       ;TEST FIRST CHAR.
002160  001015                  BNE     1$              ;BR TO ERROR
002162  122737  000057  001324          CMPB    #'/,SAVE2       ;TEST SECOND CHAR.
002170  001011                  BNE     1$              ;BR TO ERROR

                        ;NOW DETERMINE WHICH VT5XX AND ITS CHARASTICS

002172  005000                  CLR     R0              ;CLEAR INITIAL POINTER
002174  126037  002404  001326  3$:     CMPB    TYPEPT(R0),SAVE3        ;TEST I.D. TO KNOWN VALUE
002202  001406                  BEQ     4$              ;BR IF CORRECT
002204  005720                  TST     (R0)+           ;BUMP THE INDEX
002206  105760  002404          TSTB    TYPEPT(R0)      ;CHECK IF MORE VALID I.D.'S
002212  001370                  BNE     3$              ;BR IF MORE
002214  104003                  1$:     ERROR   3       ;INCORRECT I.D. CODE
002216  005000                  11$:    CLR     R0      ;DEFAULT TO VT50A MODE

                        ;HAVE NOW FOUND THE I.D. - REPORT TO CONSOLE

002220  016037  002404  001260  4$:     MOV     TYPEPT(R0),VT5XX        ;SAVE I.D. AND CHARASTICS
002226  016037  002436  002242          MOV     MSGTYP(R0),5$           ;SAVE ASCII MESSAGE POINTER
002234  001403                  BEQ     13$             ;BR IF ZERO MESSAGE POINTER
002236  004537  011442          JSR     R5,AMSG         ;TELL THE UUT
002242  016760                  5$:     VT50A           ;POINTER TO VT5XX MESSAGE

002244  012737  001700  001264  13$:    MOV     #960.,TOTALC    ;LOAD TOTAL CHARACTER COUNT
002252  012737  000014  001266          MOV     #12.,VHO        ;LOAD MAX VERTICAL LINE COUNT
002260  012737  000053  001262          MOV     #53,LASTLN      ;LOAD LAST LINE VALUE +40 FOR DCA TESTING
002266  005737  001260                  TST     VT5XX           ;TEST IF 24 LINES AVAIL
002272  100007                  BPL     6$              ;BR IF NOT
002274  006337  001264                  ASL     TOTALC          ;ADJUST CHARACTER COUNT
002300  006337  001266                  ASL     VHO             ;ADJUST LINE COUNT
```

```
                002304  062737  000014  001262          ADD     #12.,LASTLN             ;ADJUST VALID LINE # +40
                002312  013737  001266  001270  6$:     MOV     VH0,VH1                 ;LOAD OTHER LINE COUNTS
                002320  006237  001270                  ASR     VH1
                002324  013737  001270  001272          MOV     VH1,VH2
                002332  006237  001272                  ASR     VH2
                002336  012737  000140  001310          MOV     #140,LASTCH             ;LOAD FIRST NON-VALID CHARACTER
                002344  032737  004000  001260          BIT     #BIT11,VT5XX            ;TEST IF UPPER-LOWER CASE TERM.
                002352  001403                          BEQ     7$                      ;BR IF NOT
                002354  062737  000037  001310          ADD     #37,LASTCH              ;UPDATE TO ALLOW UP-LW CASE CHARACTER SET
                002362                          7$:
(1)             002362  000403                          BR      12$             ;;BR AND START TESTING
                002364  104002                  2$:     ERROR   2                       ;NO RESPONSE FROM UUT ADTER ASKING FOR IDENTIFY
                002366  000713                          BR      11$
                002370  000240                          NOP
                002372  004737  011262          12$:    JSR     PC,DELAY
                002376  000177  000000                  JMP     @WHERE
                002402  002466                  WHERE:  TST2

                                                ;I.D. VALUES AND CHARACTERISTICS
                                                ;       BIT15 = 1       24 LINES
                                                ;       BIT14 = 1       COPIER CONNECTED
                                                ;       BIT13 = 1       DIRECT CURSOR ADDRESSING (ESC Y) + "ESC-B" + "ESC-D"
                                                ;       BIT12 = 1       VT50H KEYPAD
                                                ;       BIT11 = 1       UPPER AND LOWER CASE CHARACTERS
                                                ;       BIT10 = 1       PRINTER CONNECTED
                                                ;       BIT09 = 1       VT52X MODEL

                                                ;LOW BYTE CONTAINS THE I.D. FOR EACH KNOWN VT5??

                002404  000101                  TYPEPT: .WORD   000101  ;I.D. = 101      ;VT50A
                002406  040102                          .WORD   040102  ;I.D. = 102      ;VT50B COPIER
                002410  140103                          .WORD   140103  ;I.D. = 103      ;VT55 24. LINES, COPIER(VT50 BASED)
                002412  070110                          .WORD   070110  ;I.D. = 110      ;VT50H COPIER DCA VT50H KEYPAD
                002414  135113                          .WORD   135113  ;I.D. = 113      ;VT52
                002416  175114                          .WORD   175114  ;I.D. = =114     ;VT52 WITH COPIER
                002420  137115                          .WORD   137115  ;I.D. = 115      ;VT52 WITH PRINTER.
                002422  175105                          .WORD   175105  ;I.D. = 105      ;VT55,24LINES,WITH COPIER(VT52 BASED)
                002424  000000                          0
                002426  000000                          0
                002430  000000                          0
                002432  000000                          0
                002434  000000                          0

                                                ;ASCII MESSAGE POINTERS

                002436  016760                  MSGTYP: VT50A                   ;VT50A NO COPIER
                002440  017032                          VT50B                   ;VT50B COPIER
                002442  017101                          VT55                    ;VT55 COPIER
                002444  017163                          VT50H                   ;VT50H COPIER
                002446  017241                          VT52K                   ;VT52
                002450  017324                          VT52L                   ;VT52 WITH COPIER
                002452  017405                          VT52M                   ;VT52 WITH PRINTER
                002454  017466                          VT55E
                002456  000000                          0
```

```
                                    0
                                    0
                                    0
                        ;;**********************************************************************
                        ;*TEST 2         B       FULL SCREEN OF A CHARACTER
                        ;;**********************************************************************
(2) 002466  000004      TST2:   SCOPE

002470  004537  011442            JSR     R5,AMSG
002474  013245                      M91

002476  004737  013010            JSR     PC,FILLWC       ;FILL SCREEN WITH A 'E'S

002502  004737  011262            JSR     PC,DELAY

                        ;;**********************************************************************
                        ;*TEST 3         C       DATA TRANSFER PATH TEST
                        ;;**********************************************************************
(2) 002506  000004      TST3:   SCOPE
002510  004537  011442            JSR     R5,AMSG         ;DISPLAY HEADING
002514  013313                      M92
002516  013737  001270  001312      MOV     VH1,TEMP        ;SET-UP A COUNTER

002524  004537  011154            JSR     R5,DTPSR        ;SET-UP BUFFER
002530  000077                      77                      ;OCTAL '?'
002532  000100                      100                     ;OCTAL '@'

002534  004537  011160            JSR     R5,DTPSRB       ;SET-UP BUFFER
002540  000125                      125                     ;OCTAL 'U'
002542  000052                      52                      ;OCTAL '*'

002544  004737  010236      1$:     JSR     PC,XPRNT        ;DISPLAY THIS LINE
002550  005337  001312            DEC     TEMP            ;COMPLETED FULL COUNT?
002554  001373                      BNE     1$              ;BRANCH IF NOT COMPLETED

002556  004737  011262            JSR     PC,DELAY        ;TEST DELAY SWITCH
                        ;;**********************************************************************
                        ;*TEST 4         D       SINGLE CHARACTER ACROSS ALL COLS. <ALL CHARACTERS>
                        ;;**********************************************************************
(2) 002562  000004      TST4:   SCOPE
002564  004537  011442            JSR     R5,AMSG         ;DISPLAY HEADING
002570  013342                      M93
002572  012737  000040  001306      MOV     #40,STCHAR      ;SET-UP STARTING CHARACTER

002600  013737  001266  001314      MOV     VH0,TEMPO       ;LOAD COUNT
002606  013701  001306      1$:     MOV     STCHAR,R1       ;LOAD R1= TO CHARACTER
002612  004737  010134            JSR     PC,FILBUF       ;LOAD A BUFFER WITH THAT CHARACTER

002616  004737  010236            JSR     PC,XPRNT        ;DISPLAY A FULL LINE FROM THE BUFFER

002622  005337  001314            DEC     TEMPO           ;DONE ?
002626  001005                      BNE     2$              ;FINISHED
002630  004737  011262            JSR     PC,DELAY
002634  013737  001266  001314      MOV     VH0,TEMPO
002642  005237  001306      2$:     INC     STCHAR          ;UPDATE THE CHARACTER
002646  023737  001310  001306      CMP     LASTCH,STCHAR   ;TEST FOR FINAL CHARACTER
```

```
                002654  001354                  BNE     1$              ;BRANCH IF NOT COMPLETED
                002656  004737  011262          JSR     PC,DELAY        ;TEST DELAY SWITCH
                                                ;;*********************************************************************
                                                ;*TEST 5         E       ROTATING CHARACTERS ACROSS ALL COLS. <ALL CHARACTERS>
                                                ;;*********************************************************************
(2)             002662  000004                  TST5:   SCOPE
                002664  004537  011442          JSR     R5,AMSG         ;DISPLAY HEADING
                002670  013404                  M94
                002672  012737  000040  001306  MOV     #40,STCHAR      ;SET-UP STARTING CHARACTER
                002700  013737  001266  001314  MOV     VHO,TEMPO       ;LOAD TEMP
                002706  013701  001306          1$:     MOV     STCHAR,R1       ;LOAD R1=TO CHARACTER
                002712  004537  010170          JSR     R5,LIC          ;LOAD A BUFFER STARTING WITH
                002716  001320                  WIDTH                   ; THAT CHARACTER AND WIDTH <BYTE>
                002720  004737  010236          JSR     PC,XPRNT        ;DISPLAY A FULL LINE FROM THE BUFFER
                002724  005337  001314          DEC     TEMPO           ;DONE ?
                002730  001005                  BNE     2$              ;BR IF YES
                002732  004737  011262          JSR     PC,DELAY
                002736  013737  001266  001314  MOV     VHO,TEMPO
                002744  005237  001306          2$:     INC     STCHAR          ;UPDATE THE STARTING CHARACTER
                002750  023737  001310  001306  CMP     LASTCH,STCHAR   ;TEST FOR FINAL CHARACTER
                002756  001353                  BNE     1$              ;BRANCH IF NOT COMPLETED
                002760  004737  011262          JSR     PC,DELAY        ;TEST DELAY SWITCH
                                                ;;*********************************************************************
                                                ;*TEST 6         F       CURSOR MOTION TEST
                                                ;;*********************************************************************
(2)             002764  000004                  TST6:   SCOPE
                002766  004537  011442          JSR     R5,AMSG         ;DISPLAY HEADING
                002772  013435                  M96

                                                ;ROUTINE TO LOAD REFERENCE LINES FOR CURSOR MOTION TEST

                003000  012700  025256          LBMT:   MOV     #BUFFER,R0      ;LOAD POINTER
                003004  012720  000065          MOV     #65,(R0)+       ;LOAD #5
                003010  013701  001270          MOV     VH1,R1          ;LOAD R1
                003012  005301                  DEC     R1
                003016  004737  003110          JSR     PC,MOVDN1       ;MOVE CURSOR DOWN
                003022  004737  003126          JSR     PC,MOVRIG       ;MOVE RIGHT
                003026  112720  000062          MOVB    #62,(R0)+       ;LOAD #2
                003032  004737  003126          JSR     PC,MOVRIG       ;MOVE RIGHT
                003036  004737  003126          JSR     PC,MOVRIG       ;MOVE RIGHT
                003042  112720  000040          MOVB    #40,(R0)+
                003046  112720  000061          MOVB    #61,(R0)+       ;LOAD #1
                003052  004737  003104          JSR     PC,MOVDWN       ;MOVE DOWN
                003056  004737  003126          JSR     PC,MOVRIG       ;MOVE RIGHT
                003062  112720  000063          MOVB    #63,(R0)+       ;LOAD #3
                003066  004737  003126          JSR     PC,MOVRIG
                003072  004737  003126          JSR     PC,MOVRIG       ;MOVE RIGHT
                003076  112720  000064          MOVB    #64,(R0)+       ;LOAD #4
                003102  112710  000377          MOVB    #377,(R0)       ;TERM
                003106  000420                  BR      LBMT1           ;;BR TO NEXT PART
```

```
                                                MOVDWN: MOV     VH1,R1
003104  013701  001270                          MOVDN1: MOVB    #15,(R0)+       ;LOAD CR
003110  112720  000015                                  MOVB    #12,(R0)+       ;LOAD LF
003114  112720  000012                                  DEC     R1
003120  005301                                          BNE     MOVDN1          ;LOOP UNTIL DONE
003122  001372                                          RTS     PC              ;EXIT
003124  000207

003126  012701  000024                          MOVRIG: MOV     #20.,R1         ;LOAD R1
003132  112720  000040                          1$:     MOVB    #40,(R0)+       ;LOAD SPACES
003136  005301                                          DEC     R1
003140  100374                                          BPL     1$              ;LOOP UNTIL DONE
003142  000207                                          RTS     PC              ;EXIT

003144  004737  010236                          LBMT1:  JSR     PC,XPRNT        ;DISPLAY THIS LINE
003150  004737  011262                                  JSR     PC,DELAY        ;TEST DELAY SWITCH

                                                ;CURSOR MOTION SUBROUTINE
                                                ;IF VT50H TYPE - USE "ESC-D" FOR CURSOR LEFT AND USE "ESC-B" FOR CURSOR DOWN
003154  013701  001270                          LCM:    MOV     VH1,R1          ;LOAD COUNT
003160  012700  025256                                  MOV     #BUFFER,R0      ;LOAD BUFFER POINTER
003164  112720  000033                          1$:     MOVB    #33,(R0)+       ;LOAD 'ESC'
003170  112720  000101                                  MOVB    #101,(R0)+      ;LOAD 'A' CURSOR UP
003174  005301                                          DEC     R1
003176  001372                                          BNE     1$              ;LOOP UNTIL DONE
003200  112720  000130                                  MOVB    #130,(R0)+      ;LOAD 'X'
003204  012701  000054                                  MOV     #44.,R1         ;LOAD COUNT
003210  032737  020000  001260                          BIT     #BIT13,VT5XX    ;TEST IF VT50H TYPE
003216  001407                                          BEQ     20$             ;BR IF NOT
003220  112720  000033                          2$:     MOVB    #33,(R0)+       ;LOAD 'ESC'
003224  112720  000104                                  MOVB    #104,(R0)+      ;LOAD 'CURSOR LEFT'
003230  005301                                          DEC     R1
003232  100372                                          BPL     2$              ;LOOP UNTIL DONE
003234  000404                                          BR      21$
003236  112720  000010                          20$:    MOVB    #10,(R0)+       ;LOAD "BACKSPACE
003242  005301                                          DEC     R1              ;DONE ALL ?
003244  100374                                          BPL     20$             ;BR IF NOT
003246  112720  000130                          21$:    MOVB    #130,(R0)+      ;LOAD 'X'
003252  112720  000010                                  MOVB    #10,(R0)+       ;LOAD BACKSPACE
003256  013701  001270                                  MOV     VH1,R1          ;LOAD COUNT
003262  032737  020000  001260                          BIT     #BIT13,VT5XX    ;TEST IF VT50H TYPE
003270  001407                                          BEQ     30$             ;BR IF NOT
003272  112720  000033                          3$:     MOVB    #33,(R0)+       ;LOAD 'ESC' CURSOR DOWN
003276  112720  000102                                  MOVB    #102,(R0)+      ;
003302  005301                                          DEC     R1
003304  001372                                          BNE     3$              ;LOOP UNTIL DONE
003306  000404                                          BR      31$
003310  112720  000012                          30$:    MOVB    #12,(R0)+       ;LOAD CURSOR DOWN (LF)
003314  005301                                          DEC     R1              ;DONE ?
003316  001374                                          BNE     30$             ;BR IF NOT
003320  112720  000130                          31$:    MOVB    #130,(R0)+      ;LOAD 'X'
003324  112720  000010                                  MOVB    #10,(R0)+       ;LOAD BACKSPACE
003330  012701  000052                                  MOV     #42.,R1         ;LOAD COUNT
003334  112720  000033                          4$:     MOVB    #33,(R0)+       ;LOAD 'ESC'
003340  112720  000103                                  MOVB    #103,(R0)+      ;LOAD 'C' CURSOR RIGHT
003344  005301                                          DEC     R1
```

```
1036  003346  100372                  BPL     4$                      ;LOOP UNTIL DONE
1037  003350  112720  000130          MOVB    #130,(R0)+              ;LOAD 'X'
1038  003354  112720  000033          MOVB    #33,(R0)+               ;LOAD 'ESC'
1039  003360  112720  000110          MOVB    #110,(R0)+              ;LOAD 'H' CURSOR HOME
1040  003364  112720  000130          MOVB    #130,(R0)+
1041  003370  112720  000377          MOVB    #377,(R0)+
1042  003374  004737  010236          JSR     PC,XPRNT                ;DISPLAY THIS LINE
1043  003400  004737  011262          JSR     PC,DELAY                ;DELAY
1044  003404  004537  011442          JSR     R5,AMSG                 ;
1045  003410  016463                  CRLFB
1046  003412  005737  001260          TST     VTSXX                   ;TEST IF 24 LINES
1047  003416  100003                  BPL     TST7                    ;BR IF 12 LINES
1048  003420  004537  011442          JSR     R5,AMSG                 ;SCROLL MORE LINES
1049  003424  016463                  CRLFB
1050
1051
                        ;;*****************************************************************
                        ;*TEST 7         G       TAB, BACKSPACE AND BELL TEST
                        ;;*****************************************************************
(3)
(3)
(2)   003426  000004    TST7:   SCOPE
1056  003430  004537  011442          JSR     R5,AMSG                 ;DISPLAY HEADING
1057  003434  013470                  M97
1058
1059  003436  012700  025256  LRL:    MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
1060  003442  112720  000007          MOVB    #7,(R0)+                ;LOAD 'BELL'
1061  003446  012701  000011          MOV     #9.,R1                  ;LOAD LINE COUNT
1062  003452  012702  000006  1$:     MOV     #6.,R2                  ;LOAD COLUMN COUNT
1063  003456  112720  000111          MOVB    #111,(R0)+              ;LOAD COLUMN MARK
1064  003462  112720  000040  2$:     MOVB    #40,(R0)+               ;LOAD SPACE
1065  003466  005302                  DEC     R2                      ;
1066  003470  100374                  BPL     2$                      ;BR UNTIL DONE
1067  003472  005301                  DEC     R1                      ;
1068  003474  100366                  BPL     1$                      ;BR UNTIL FINISHED ALL TAB STOPS
1069  003476  112720  000015          MOVB    #15,(R0)+               ;LOAD CR
1070  003502  112720  000012          MOVB    #12,(R0)+               ;LOAD LF
1071  003506  112720  000377          MOVB    #377,(R0)+              ;LOAD TERM
1072
1073  003512  004737  010236          JSR     PC,XPRNT
1074
1075  003516  004537  003706          JSR     R5,LTAB                 ;LOAD TAB LINE #1
1076  003522  000000                  0
1077  003524  004737  010236          JSR     PC,XPRNT                ;DISPLAY THIS LINE
1078
1079  003530  004537  003706          JSR     R5,LTAB                 ;LOAD TAB LINE #2
1080  003534  000001                  1
1081  003536  004737  010236          JSR     PC,XPRNT                ;DISPLAY THIS LINE
1082
1083  003542  004537  003706          JSR     R5,LTAB                 ;LOAD TAB LINE #3
1084  003546  000002                  2
1085  003550  004737  010236          JSR     PC,XPRNT                ;DISPLAY THIS LINE
1086
1087  003554  004537  003706          JSR     R5,LTAB                 ;LOAD TAB LINE #4
1088  003560  000003                  3
1089  003562  004737  010236          JSR     PC,XPRNT                ;DISPLAY THIS LINE
1090
1091  003566  004537  003706          JSR     R5,LTAB                 ;LOAD TAB LINE #5
1092  003572  000004                  4
```

```
1089  003574  004737  010236                    JSR     PC,XPRNT        ;DISPLAY THIS LINE
1090
1091  003600  004537  003706                    JSR     R5,LTAB         ;LOAD TAB LINE #6
1092  003604  000005                            5
1093  003606  004737  L10236                    JSR     PC,XPRNT        ;DISPLAY THIS LINE
1094
1095  003612  004537  003706                    JSR     R5,LTAB         ;LOAD TAB LINE #7
1096  003616  000006                            6
1097  003620  004737  010236                    JSR     PC,XPRNT        ;DISPLAY THIS LINE
1098                                    ;NOW EXECUTE THE BACKSPACE SECTION
1099
1100  003624  013702  001320                    MOV     WIDTH,R2        ;LOAD WIDTH COUNT
1101  003630  005302                            DEC     R2              ;ADJUST WIDTH
1102  003632  005302                            DEC     R2              ;  BY 2
1103  003634  012701  000040                    MOV     #40,R1          ;LOAD "SPACE" INTO THE LINE
1104  003640  004737  010140                    JSR     PC,FILBFB       ;LOAD LINE
1105  003644  013701  001320                    MOV     WIDTH,R1        ;LOAD # OF CHARACTER POSITIONS
1106  003650  112720  000130            3$:     MOVB    #'X,(R0)+       ;LOAD ASCII "X"
1107  003654  112720  000010                    MOVB    #10,(R0)+       ;LOAD BACKSPACE CODE
1108  003660  112720  000010                    MOVB    #10,(R0)+
1109  003664  005301                            DEC     R1              ;DONE ALL POSITIONS ?
1110  003666  001370                            BNE     3$              ;BR IF NOT
1111  003670  112720  000377                    MOVB    #377,(R0)+      ;LOAD TERM.
1112  003674  004737  010236                    JSR     PC,XPRNT        ;EXECUTE ASCII CODE
1113  003700  004737  011262                    JSR     PC,DELAY        ;TEST DELAY SWITCH
1114  003704  000434                            BR      TST10           ;;BR TO NEXT TEST
1115
1116                                    ;SUBROUTINE TO LOAD THE TAB TEST INTO THE BUFFER
1117
1118  003706  012537  001322            LTAB:   MOV     (R5)+,SAVE1     ;LOAD # OF CHARACTERS
1119  003712  012702  025256                    MOV     #BUFFER,R2      ;LOAD BUFFER POINTER
1120  003716  012701  000011                    MOV     #9.,R1          ;LOAD WIDTH COUNTER
1121  003722  112722  000007                    MOVB    #7,(R2)+        ;LOAD BELL AT START OF LINE
1122  003726  112722  000111            3$:     MOVB    #11,(R2)+       ;LOAD 'I
1123  003732  013700  001322                    MOV     SAVE1,R0        ;GET THE  NO. OF THE CHAR
1124  003736  001404                            BEQ     1$              ;BR IF 0
1125  003740  112722  000101            2$:     MOVB    #101,(R2)+      ;LOAD THE 'A' CHAR.
1126  003744  005300                            DEC     R0              ;
1127  003746  001374                            BNE     2$              ;LOOP UNTIL DONE
1128  003750  112722  000011            1$:     MOVB    #11,(R2)+       ;LOAD 'TAB' CHAR
1129  003754  005301                            DEC     R1              ;
1130  003756  100363                            BPL     3$              ;BR TO NEXT TAB COLUMN CHAR
1131  003760  112722  000015                    MOVB    #15,(R2)+       ;LOAD 'CR'
1132  003764  112722  000012                    MOVB    #12,(R2)+       ;LOAD 'LF'
1133  003770  112722  000377                    MOVB    #377,(R2)+      ;LOAD TERM
1134  003774  000205                            RTS     R5              ;EXIT
1135                            ;;*****************************************************************
  (3)                           ;*TEST 10       H       ERASE FROM CURSOR TO END OF LINE
  (3)                           ;*****************************************************************
  (2)  003776  000004            TST10:  SCOPE
1136  004000  004537  011442                    JSR     R5,AMSG         ;DISPLAY HEADING
1137  004004  013536                            M910
1138
1139  004006  004737  013010                    JSR     PC,FILLWC       ;FILL BUFFER WITH A CHAR
1140                                                                    ;DISPLAY THIS LINE
1141  004012  004737  011262                    JSR     PC,DELAY        ;DELAY
```

```
                                        ;LOAD ERASE LINE TEST INTO BUFFER
004016  012700  025256          LODERL: MOV     #BUFFER,R0
004022  112720  000033                  MOVB    #33,(R0)+       ;LOAD ESC
004026  112720  000110                  MOVB    #'H,(R0)+       ;LOAD HOME
004032  013737  001266  004134          MOV     VHO,2$          ;LOAD COUNT
004040  005337  004134                  DEC     2$
004044  112720  000033          3$:     MOVB    #33,(R0)+       ;LOAD ESC
004050  112720  000113                  MOVB    #'K,(R0)+       ;LOAD ERASE LINE CHAR
004054  005337  004134                  DEC     2$              ;FINISHED ?
004060  100427                          BMI     1$              ;BR WHEN DONE
004062  012737  000006  004136          MOV     #6.,10$         ;LOAD COUNT
004070  005737  001260                  TST     VTSXX           ;TEST IF 12 LINES
004074  100005                          BPL     4$              ;BR IF 12 LINES
004076  006237  004136                  ASR     10$             ;ADJUST HORIZ. COUNT
004102  000240                          NOP
004104  000240                          NOP
004106  000240                          NOP
004110  112720  000033          4$:     MOVB    #33,(R0)+
004114  112720  000103                  MOVB    #'C,(R0)+       ;CURSOR RIGHT
004120  005337  004136                  DEC     10$             ;
004124  001371                          BNE     4$              ;LOOP
004126  112720  000012                  MOVB    #12,(R0)+       ;CURSOR DOWN
004132  000744                          BR      3$
004134  000000          2$:     0
004136  000000          10$:    0
004140  112720  000377          1$:     MOVB    #377,(R0)+      ;LOAD TERM

004144  004737  010236                  JSR     PC,XPRNT        ;DISPLAY THIS TEST

004150  004737  011262                  JSR     PC,DELAY        ;TEST DELAY SWITCH

                                ;;**************************************************************
                                ;*TEST 11        I       ERASE FROM CURSOR TO END OF SCREEN
                                ;;**************************************************************
004154  000004                  ST11:   SCOPE
004156  004537  011442                  JSR     R5,AMSG         ;DISPLAY HEADING
004162  013567                          M911
004164  004737  013010                  JSR     PC,FILLWC       ;FILL BUFFER WITH A CHAR
                                                                ;DISPLAY THIS LINE
004170  004737  011262                  JSR     PC,DELAY        ;DELAY

                                ;LOAD ERASE SCREEN TEST INTO BUFFER

004174  013737  001320  004270  LODERS: MOV     WIDTH,2$        ;LOAD COUNT
004202  006237  004270                  ASR     2$
004206  012700  025256                  MOV     #BUFFER,R0
004212  112720  000010          1$:     MOVB    #10,(R0)+       ;LOAD CURSOR LEFT
004216  005337  004270                  DEC     2$              ;DONE ?
004222  100373                          BPL     1$              ;BR UNTIL DONE
004224  013737  001266  004270          MOV     VHO,2$          ;LOAD COUNT
004232  112720  000033          4$:     MOVB    #33,(R0)+       ;LOAD ESC
004236  112720  000112                  MOVB    #'J,(R0)+       ;LOAD ERASE SCREEN
004242  005337  004270                  DEC     2$              ;DONE ?
```

```
1195  004246  100405                          BMI     3$                      ;BR WHEN DONE
1196  004250  112720  000033                  MOVB    #33,(R0)+               ;LOAD ESC
1197  004254  112720  000101                  MOVB    #'A,(R0)+               ;LOAD CURSOR UP
1198  004260  000764                          BR      4$                      ;LOOP
1199  004262  112720  000377          3$:     MOVB    #377,(R0)+              ;LOAD TERM
1200  004266  000401                          BR      5$
1201  004270  000000                  2$:     0
1202
1203  004272  004737  010236          5$:     JSR     PC,XPRNT                ;DISPLAY THIS TEST
1204  004276  004737  011262                  JSR     PC,DELAY                ;TEST DELAY SWITCH
1205
1206                                  ;;**********************************************************************
  (3)                                 ;*TEST 12        J       VIDEO COUPLING TEST
  (3)                                 ;;**********************************************************************
  (2)  004302  000004                 TST12:  SCOPE
1207  004304  004537  011442                  JSR     R5,AMSG                 ;DISPLAY HEADER
1208  004310  013622                          M912
1209  004312  013737  001270  001312          MOV     VH1,TEMP                ;LOAD COUNTER
1210  004320  004537  011442          1$:     JSR     R5,AMSG
1211  004324  016511                          PATH
1212
1213  004326  005337  001312                  DEC     TEMP                    ;FINISHED COUNT ?
1214  004332  001372                          BNE     1$                      ;BR UNTIL DONE
1215  004334  004737  011262                  JSR     PC,DELAY                ;TEST DELAY SWITCH
1216
1217                                  ;;**********************************************************************
  (3)                                 ;*TEST 13        K       DIRECT CURSOR ADDRESS TEST
  (3)                                 ;;**********************************************************************
  (2)  004340  000004                 TST13:  SCOPE
1218                                  ;       RANDOM NUMBERS ARE GENERATED AND USED AS "X" AND "Y" COORDINATES
1219                                  ;ADDRESSING A 960 CHARACTER PRINTOUT.
1220                                  ;VERFICATION OF DISPLAY IS PERFORMED VISUALLY BY THE USER
1221                                  ;EXECUTE 1ST TIME USING "ESC-Y" SEQUENCE AND IF I.D. IS AN "H"
1222                                  ;       EXECUTE 2ND TIME USING "CODE 16" SEQUENCE
1223
1224  004342  032737  020000  001260          BIT     #BIT13,VT5XX            ;TEST IF DCA TYPE TERMINAL
1225  004350  001002                          BNE     3$                      ;;BR IF CURSOR ADDRESSING
1226  004352  000137  005100          1$:     JMP     TST14                   ;BYPASS THIS TEST
1227  004356  005737  001332          3$:     TST     WFTEST                  ;TEST IF IN W.F. MODE
1228  004362  001373                          BNE     1$                      ;BYPASS IF W.F. MODE
1229  004364  004537  011442                  JSR     R5,AMSG                 ;ERASE SCREEN AND IDENTIFY TEST
1230  004370  020110                          M98
1231  004372  112737  000033  025256          MOVB    #33,BUFFER              ;LOAD "ESC" CODE
1232  004400  112737  000131  025257          MOVB    #'Y,BUFFER+1            ;LOAD ASCII "Y" (D.C.A. ENABLE)
1233  004406  004737  004462                  JSR     PC,DCATST               ;RUN TEST WITH ESC Y SEQUENCE
1234  004412  004737  011262                  JSR     PC,DELAY                ;DELAY TESTING
1235  004416  122737  000110  001260          CMPB    #'H,VT5XX               ;TEST IF VT50H TYPE DISPLAY
1236  004424  001352                          BNE     1$                      ;;BR IF NOT VT50H TYPE
1237  004426  004537  011442                  JSR     R5,AMSG                 ;ERASE SCREEN AND IDENTIFY TEST
1238  004432  020110                          M98
1239  004434  112737  000000  025256          MOVB    #0,BUFFER               ;LOAD "SPACE" AS FIRST VALUE
1240  004442  112737  000016  025257          MOVB    #16,BUFFER+1            ;LOAD CODE-16 SEQUENCE
1241  004450  004737  004462                  JSR     PC,DCATST               ;RUN TEST WITH CODE 16 SEQUENCE
1242  004454  004737  011262                  JSR     PC,DELAY
1243  004460  000734                          BR      1$                      ;;BR TO NEXT TEST
1244
```

```
 1245  004462  012737  123456  024756  DCATST: MOV     #123456,$LONUM  ;PRIME RANDOM NUMBER GENERATOR
 1246  004470  012737  176543  024754          MOV     #176543,$HINUM
 1247  004476  013737  001264  004754          MOV     TOTALC,OVRAL    ;LOAD CHAR COUNT
 1248  004504  013737  001266  004752          MOV     VHO,SET         ;LOAD LINE COUNT
 1249  004512  012700  025276                  MOV     #BUFFER+20,R0   ;LOAD DESTINATION BUFFER
 1250  004516  012701  004756          2$:     MOV     #MSGTXT,R1      ;LOAD MESSAGE POINTER
 1251  004522  012120                  1$:     MOV     (R1)+,(R0)+     ;LOAD 2 CHARACTERS
 1252  004524  022701  005076                  CMP     #MSGTND,R1      ;TEST FOR LAST CHAR
 1253  004530  001374                          BNE     1$              ;BR UNTIL DONE 1 LINE
 1254  004532  005337  004752                  DEC     SET             ;FINISHED ALL LINES
 1255  004536  001367                          BNE     2$              ;BR IF NOT
 1256  004540  012737  177777  025262          MOV     #-1,BUFFER+4    ;LOAD MESSAGE TERMINATOR
 1257
 1258  004546  004737  024656          GENER:  JSR     %7,$RAND        ;GENERATE RANDOM NUMBER
 1259  004552  013700  024754                  MOV     $HINUM,R0       ;GET RNADOM NUMBER
 1260  004556  042700  177700                  BIC     #177700,%0      ;RANDOM NO, MUST BE TWO DIGITS
 1261  004562  020027  000037                  CMP     %0,#37          ;NO, MUST BE LESS THAN 40
 1262  004566  101767                          BLOS    GENER           ;LOWER, REGENERATION
 1263  004570  020037  001262                  CMP     %0,LASTLN               ;NO, MUST NOT BE GREATER THAN 54 OR 67
 1264  004574  101364                          BHI     GENER           ;GREATER, REGENERATION
 1265  004576  010037  004746                  MOV     %0,YADDS        ;STORE RANDOM Y COORDINATE
 1266  004602  010001                          MOV     %0,%1           ;COPY DATA
 1267  004604  012737  025276  004752          MOV     #BUFFER+20,SET  ;LOAD BASE POINTER
 1268  004612  162701  000040                  SUB     #40,%1          ;MINIMUM Y INDEX
 1269  004616  001405                          BEQ     GENRX           ;RESULT, MINIMUM Y COORDINATE
 1270  004620  062737  000120  004752  1$:     ADD     #80.,SET        ;SETUP 'Y INDEX LOCATION FOR PRINTOUT
 1271  004626  005301                          DEC     %1
 1272  004630  001373                          BNE     1$              ;Y COORDINATE IS SET
 1273  004632  004737  024656          GENRX:  JSR     %7,$RAND        ;GENERATE RANDOM NUMBER
 1274  004636  013700  024754                  MOV     $HINUM,R0       ;GET A RANDOM NUMBER
 1275  004642  042700  177600                  BIC     #177600,%0      ;RANDOM NO. MAY BE LESS THAN 200
 1276  004646  020027  000037                  CMP     %0,#37          ; MUST NOT BE LESS THAN 40
 1277  004652  101767                          BLOS    GENRX           ;LOWER, REGENERATION
 1278  004654  020027  000157                  CMP     %0,#157         ; MUST NOT BE GREATER THAN 157
 1279  004660  101364                          BHI     GENRX           ;GREATER, REGENERATION
 1280  004662  010037  004750                  MOV     %0,XADDS        ;STORE RANDOM X COORDINATE
 1281  004666  162700  000040                  SUB     #40,%0          ;SETUP MINIMUM X INDEX
 1282  004672  060037  004752                  ADD     %0,SET          ;SETUP X COOR, FOR PNTOUT.
 1283  004676  013701  004752                  MOV     SET,%1          ;SETUP CHECK
 1284  004702  105711                          TSTB    (1)             ;HAS CURRENT CHAR, ALREADY BEEN USED?
 1285  004704  100720                          BMI     GENER           ;YES, REGENERATE
 1286  004706  113737  004746  025260          MOVB    YADDS,BUFFER+2  ;LOAD Y COORDINATE
 1287  004714  113737  004750  025261          MOVB    XADDS,BUFFER+3  ;LOAD X COORDINATE
 1288  004722  111137  025262                  MOVB    (1),BUFFER+4    ;LOAD CHARACTER TO BE PRINTED
 1289  004726  152711  000200                  BISB    #200,(1)        ;INDICATE USE OF CURSOR POSITION
 1290  004732  004737  010236                  JSR     PC,XPRNT        ;EXECUTE AND PRINT CHARACTER
 1291  004736  005337  004754                  DEC     OVRAL           ;MAXIMUM NO. OF COORDINATES
 1292  004742  001301                          BNE     GENER           ;BR BACK UNTIL DONE
 1293  004744  000207                          RTS     PC              ;EXIT
 1294
 1295  004746  000000                  YADDS:  0
 1296  004750  000000                  XADDS:  0
 1297  004752  000000                  SET:    0
 1298  004754  000000                  OVRAL:  0
 1299  004756  052126  030065  050055  MSGTXT: .ASCII  \VT50-PLUS-DIRECT-CURSOR-ADDRESSING-TEST\
       004764  052514  026523  044504
```

```
                004772  042522  052103  041455
                005000  051125  047523  026522
                005006  042101  051104  051505
                005014  044523  043516  052055
                005022  051505  000124
1300            005025  000055  044504  044507          .ASCII  \-DIGITAL-EQUIPMENT-CORP.-MAYNARD-MA.-VT50\
                005032  040524  026514  050505
                005040  044525  046520  047105
                005046  026524  047503  050122
                005054  026456  040515  047131
                005062  051101  026504  040515
                005070  026456  052126  030065
1301            005076  100000                  MSGTND: BIT15
1302                                            ;ONLY 12 LINE TERMINALS WILL RUN THIS TEST
1303                                            ;;*****************************************************************
  (3)                                           ;*TEST 14        L       HOLD SCREEN TEST
  (3)                                           ;;*****************************************************************
  (2)           005100  000004                  TST14:  SCOPE
1304            005102  005037  010656                  CLR     AXOFF
1305            005106  005037  010660                  CLR     XOFFRC          ;CLEAR SOFT FLAG
1306            005112  004537  011442                  JSR     R5,AMSG         ;DISPLAY
1307            005116  016463                          CRLFB
1308            005120  004537  011442                  JSR     R5,AMSG         ;CR-LF
1309            005124  016463                          CRLFB
1310
1311            005126  005737  001332                  TST     WFTEST          ;TEST IF IN W.F. MODE
1312            005132  001046                          BNE     TST15           ;;BR IF W.F. MODE
1313            005134  005737  001260                  TST     VT5XX           ;TEST IF 12 LINE
1314            005140  100443                          BMI     TST15           ;;BR IF NOT 12 LINE
1315            005142  004537  011442                  JSR     R5,AMSG         ;DISPLAY MESSAGE
1316            005146  014007                          M922
1317
1318            005150  012737  000001  010654          MOV     #1,XOFFOK
1319            005156  012737  000001  010652          MOV     #1,XOFFBR       ;ENABLE XOFF
1320            005164  004537  011442                  JSR     R5,AMSG         ;TRY TO  SCROLL THE SCREEN
1321            005170  020000                          GOHDSC                  ;ENABLE HOLD SCREEN
1322
1323            005172  005737  010660                  TST     XOFFRC          ;TEST IF XOFF SENSED
1324            005176  001001                          BNE     1$              ;BR IF SENSED
1325
1326            005200  104002                          ERROR   2               ;ENABLE HOLD SCREEN MODE FAILED TO
1327                                                                            ;INHIBIT THE SCREEN FROM SCROLLING
1328                                                                            ;BY SENDING "X-OFF"
1329
1330            005202  005037  010662          1$:     CLR     XONRC           ;CLEAR SOFT FLAG
1331            005206  005037  010652                  CLR     XOFFBR
1332            005212  012737  000001  010656          MOV     #1,AXOFF
1333            005220  004537  011442                  JSR     R5,AMSG
1334            005224  020026                          GODSHS                  ;DISABLE HOLD SCREEN MODE
1335
1336            005226  004537  011442                  JSR     R5,AMSG
1337            005232  016474                          CRLFA                   ;TRY SCROLLING THE SCREEN
1338
1339            005234  005737  010662                  TST     XONRC           ;TEST SOFT FLAG (X-ON)
1340            005240  001001                          BNE     2$              ;BR IF SENSED
1341
```

```
                                                ERROR   2               ;DISABLE HOLD SCREEN MODE FAILED TO ENABLE
                                                                        ;THE SCREEN TO SCROLL BY SENDING AN "X-ON"
                                        2$:     JSR     PC,DELAY        ;PROGRAM DELAY
                                        ;;*****************************************************************
                                        ;*TEST 15       M       TEST GRAPHICS MODE AND REV. LINE FEED
                                        ;;*****************************************************************
                                        TST15:  SCOPE
005252  032737  001000  001260                  BIT     #BIT09,VT5XX    ;IS UNIT VT52?
005260  001002                                  BNE     GRPHST          ;YES-TEST GRAPHICS AND REV. LINE FEED
005262  000137  005362                          JMP     COPTST          ;NO-GO CHECK FOR COPIER
005266  012704  000120                  GRPHST: MOV     #80.,R4
005272  004537  011442                  1$:     JSR     R5,AMSG         ;HOME THE CURSOR AND CLEAR THE SCREEN.
005276  016503                                  HOMERS
005300  004537  011442                          JSR     R5,AMSG         ;DISPLAY TEST MESSAGE
005304  014044                                  M9221
005306  004537  011442                          JSR     R5,AMSG         ;ENTER GRAPHICS MODE.
005312  021555                                  ENGRAF
005314  012737  000045  001314                  MOV     #37.,TEMPO      ;SET UP TO XMIT 80 LINES
005322  004737  011220                  2$:     JSR     PC,GBBUF        ;LOAD BUFFER WITH GRAPHICS
005326  004737  010236                          JSR     PC,XPRNT        ;DISPLAY 1 LINE
005332  004537  011442                          JSR     R5,AMSG         ;ISSUE REV. LINE FEED AND
005336  021536                                  REVLF                   ;A CARRIAGE RETURN
005340  005304                                  DEC     R4              ;DECREMENT BUFFER COUNT
005342  005337  001314                  3$:     DEC     TEMPO           ;DONE?
005346  001365                                  BNE     2$              ;NO-LOOP
005350  004537  011442                          JSR     R5,AMSG         ;YES-EXIT GRAPHICS MODE
005354  021562                                  EXGRAF
005356  004737  011262                          JSR     PC,DELAY        ;AND GO CHECK DELAY
005362                                  COPTST:
                                        ;;*****************************************************************
                                        ;*TEST 16       N       COPIER - AUTO COPY TEST
                                        ;;*****************************************************************
005362  000004                          TST16:  SCOPE
005364  032737  040000  001260                  BIT     #BIT14,VT5XX    ;TEST IF COPIER IS AVAILABLE
005372  001002                                  BNE     2$              ;;BR IF AVAILABLE
005374  000137  006342                  3$:     JMP     PRNTST          ;NOT AVAILABLE SO BYPASS COPIER TESTS
005400  032777  004000  173532          2$:     BIT     #BIT11,@SWR     ;TEST IF INHIBIT COPIER TEST SWITCH IS SET
005406  001372                                  BNE     3$              ;BR IF SET
005410  032777  002000  173522                  BIT     #BIT10,@SWR     ;TEST IF PAPER SAVE SWITCH IS SET
005416  001406                                  BEQ     6$              ;BR IF NOT SET AND START COPIER TESTING
005420  105737  001274                          TSTB    PRTCNT          ;TEST IF TIME TO TEST COPIER
005424  001363                                  BNE     3$              ;NOT TIME TO RUN THE COPIER
005426  013737  001250  001274                  MOV     PTCT,PRTCNT     ;RELOAD COUNTER AND TEST COPIER

005434  004537  011442                  6$:     JSR     R5,AMSG
005440  014121                                  M923                    ;DISPLAY HEADER
005442  004537  011442                          JSR     R5,AMSG         ;ENABLE AUTO-COPY
005446  020054                                  GOAPMD

005450  013737  001266  001314                  MOV     VHO,TEMPO       ;LOAD A EXECUTION COUNT
005456  012701  000101                          MOV     #101,R1         ;LOAD A CHARACTER
005462  004737  010134                          JSR     PC,FILBUF       ;LOAD CHARACTER INTO BUFFER
005466  012737  000001  010654          1$:     MOV     #1,XOFFOK
005474  004737  010236                          JSR     PC,XPRNT        ;DISPLAY IT
```

```
1391  005500  004737  011262                    JSR     PC,DELAY        ;PROGRAM DELAY
1392  005504  005337  001314                    DEC     TEMPO           ;FINISHED ?
1393  005510  100366                            BPL     1$              ;BR IF NOT
1394  005512  012737  000001  010654            MOV     #1,XOFFOK
1395  005520  004537  011442                    JSR     R5,AMSG         ;DISABLE AUTO-COPY
1396  005524  020061                            GONAPM
1397  005526  004737  011262                    JSR     PC,DELAY        ;ANOTHER DELAY
1398
1399
  (3)                                   ;;*****************************************************************
  (3)                                   ;*TEST 17        0       COPIER - FULL SCREEN OF A CHARACTER
  (3)                                   ;;*****************************************************************
  (2)  005532  000004                   TST17:  SCOPE
1400  005534  004537  011442            1$:     JSR     R5,AMSG         ;DISPLAY HEADER
1401  005540  013245                            M91
1402  005542  004737  013010                    JSR     PC,FILLWC       ;FILL BUFFER WITH CHAR
1403                                                                    ;DISPLAY IT
1404  005546  012737  000001  010654            MOV     #1,XOFFOK       ;ALLOW XOFF
1405  005554  004537  011442                    JSR     R5,AMSG         ;COPY INST
1406  005560  017752                            GOPRNT
1407  005562  004737  011262                    JSR     PC,DELAY
1408
1409
1410                                   ;;*****************************************************************
  (3)                                   ;*TEST 20        P       COPIER - DATA PATH TEST
  (3)                                   ;;*****************************************************************
  (2)  005566  000004                   TST20:  SCOPE
1411  005570  004537  011442                    JSR     R5,AMSG         ;DISPLAY HEADING
1412  005574  013313                            M92
1413  005576  013737  001270  001312            MOV     VH1,TEMP        ;SET-UP A COUNTER
1414  005604  004537  011154                    JSR     R5,DTPSR        ;SET-UP BUFFER
1415  005610  000077                            77                      ;OCTAL '?'
1416  005612  000100                            100                     ;OCTAL '@'
1417  005614  004537  011160                    JSR     R5,DTPSRB       ;SET-UP BUFFER
1418  005620  000125                            125                     ;OCTAL 'U'
1419  005622  000052                            52                      ;OCTAL '*'
1420  005624  004737  010236            1$:     JSR     PC,XPRNT        ;DISPLAY THIS LINE
1421  005630  005337  001312                    DEC     TEMP            ;COMPLETED FULL COUNT?
1422  005634  001373                            BNE     1$              ;BRANCH IF NOT COMPLETED
1423  005636  012737  000001  010654            MOV     #1,XOFFOK
1424  005644  004537  011442                    JSR     R5,AMSG
1425  005650  017752                            GOPRNT                  ;COPY INST
1426  005652  004737  011262                    JSR     PC,DELAY        ;TEST DELAY SWITCH
1427
1428                                   ;;*****************************************************************
  (3)                                   ;*TEST 21        Q       COPIER - SINGLE CHARACTER ACROSS ALL COLS. <ALL CHARACTERS>
  (3)                                   ;;*****************************************************************
  (2)  005656  000004                   TST21:  SCOPE
1429  005660  004537  011442                    JSR     R5,AMSG         ;DISPLAY HEADING
1430  005664  013342                            M93
1431  005666  012737  000040  001306            MOV     #40,STCHAR      ;SET-UP STARTING CHARACTER
1432  005674  013737  001266  001314            MOV     VHQ,TEMPQ
1433  005702  013701  001306            1$:     MOV     STCHAR,R1       ;LOAD R1= TO CHARACTER
1434  005706  004737  010134                    JSR     PC,FILBUF       ;LOAD A BUFFER WITH THAT CHARACTER
1435  005712  004737  010236                    JSR     PC,XPRNT        ;DISPLAY A FULL LINE FROM THE BUFFER
1436  005716  005337  001314                    DEC     TEMPQ           ;DONE
1437  005722  001011                            BNE     3$
```

```
                                    MOV     VHO,TEMPO
                                    MOV     #1,XOFFOK       ;ALLOW XOFF
                                    JSR     R5,AMSG
                                    GOPRNT                  ;COPY INST
                            3$:     INC     STCHAR          ;UPDATE THE CHARACTER
                                    CMP     LASTCH,STCHAR   ;TEST FOR FINAL CHARACTER
                                    BNE     1$              ;BRANCH IF NOT COMPLETED
                                    JSR     R5,AMSG         ;CRLF
                                    CRLFA-2
                                    MOV     #1,XOFFOK       ;ENABLE XOFF
                                    JSR     R5,AMSG         ;COPY INS
                                    GOPRNT
                                    JSR     PC,DELAY        ;TEST DELAY SWITCH

                            ;;*******************************************************************
                            ;*TEST 22        R       COPIER - ROTATING CHARACTERS ACROSS ALL COLS. <ALL CHARACTERS>
                            ;;*******************************************************************
                            TST22:  SCOPE
                                    JSR     R5,AMSG         ;DISPLAY HEADING
                                    M94
                                    MOV     #40,STCHAR      ;SET-UP STARTING CHARACTER
                                    MOV     VHO,TEMPO       ;
                            1$:     MOV     STCHAR,R1       ;LOAD R1=TO CHARACTER
                                    JSR     R5,LIC          ;LOAD A BUFFER STARTING WITH
                                    WIDTH                   ;  THAT CHARACTER AND WIDTH <BYTE>
                                    JSR     PC,XPRNT        ;DISPLAY A FULL LINE FROM THE BUFFER
                                    DEC     TEMPO           ;DONE ?
                                    BNE     3$
                                    MOV     VHO,TEMPO
                                    MOV     #1,XOFFOK
                                    JSR     R5,AMSG
                                    GOPRNT                  ;COPY INST
                            3$:     INC     STCHAR          ;UPDATE THE STARTING CHARACTER
                                    CMP     LASTCH,STCHAR   ;TEST FOR FINAL CHARACTER
                                    BNE     1$              ;BRANCH IF NOT COMPLETED
                                    JSR     R5,AMSG         ;CRLF
                                    CRLFA-2
                                    MOV     #1,XOFFOK       ;ENABLE XOFF
                                    JSR     R5,AMSG         ;COPY INST
                                    GOPRNT
                                    JSR     PC,DELAY        ;TEST DELAY SWITCH

                            ;;*******************************************************************
                            ;*TEST 23        S       COPIER - PERIMETER PATTERN
                            ;;*******************************************************************
                            TST23:  SCOPE
                                    JSR     R5,AMSG         ;DISPLAY HEADER
                                    M920
                                    MOV     #'E,R1          ;LOAD STARTING CHARACTER
                                    JSR     PC,FILBUF       ;FILL THE BUFFER
                                    JSR     PC,XPRNT        ;DISPLAY A LINE
                                    JSR     PC,XPRNT        ;DISPLAY A LINE
                                    MOV     VH1,TEMP        ;LOAD VERT COUNT
                                    ADD     #2,TEMP         ;UPDATE BY 2
                            1$:     JSR     PC,FIRLST       ;FILL FIRST AND LAST
                                    JSR     PC,XPRNT        ;DISPLAY A LINE
```

```
1488  006220  005337  001312                  DEC     TEMP            ;DONE ?
1489  006224  001371                          BNE     1$
1490  006226  012701  000105                  MOV     #'E,R1          ;LOAD STARTING CHAR
1491  006232  004737  010134                  JSR     PC,FILBUF       ;LOAD LINE WITH E'S
1492  006236  004737  010236                  JSR     PC,XPRNT        ;DISPLAY A LINE
1493  006242  004737  010236                  JSR     PC,XPRNT
1494  006246  012737  000001  010654          MOV     #1,XOFFOK
1495  006254  004537  011442                  JSR     R5,AMSG
1496  006260  017752                          GOPRNT                  ;COPY SCREEN
1497  006262  004737  011262                  JSR     PC,DELAY
1498
1499                                  ;;*******************************************************************
  (3)                                 ;*TEST 24        T       COPIER - DISCLAIMER STATEMENT
  (3)                                 ;;*******************************************************************
  (2)  006266  000004                 TST24:  SCOPE
1500
1501  006270  004537  011442                  JSR     R5,AMSG         ;DISPLAY HEADER
1502  006274  013753                          M921
1503
1504  006276  005004                          CLR     R4
1505  006300  016437  006616  006314  1$:     MOV     DISPCH(R4),10$  ;LOAD A POINTER
1506  006306  001405                          BEQ     2$              ;BR IF DONE
1507  006310  004537  011442                  JSR     R5,AMSG         ;DISPLAY COPYRIGHT
1508  006314  022131                  10$:    MTEXTO
1509  006316  005724                          TST     (R4)+           ;UPDATE POINTER
1510  006320  000767                          BR      1$
1511
1512
1513  006322  012737  000001  010654  2$:     MOV     #1,XOFFOK       ;ENABLE XOFF
1514  006330  004537  011442                  JSR     R5,AMSG         ;COPY SCREEN
1515  006334  017752                          GOPRNT
1516  006336  004737  011262                  JSR     PC,DELAY
1517  006342  000240                  PRNTST: NOP                     ;TRY PRINTER TESTS
1518
1519
                                    ;;*******************************************************************
  (3)                                 ;*TEST 25        U       PRINTER CONTROLLER MODE TEST
  (3)                                 ;;*******************************************************************
  (2)  006344  000004                 TST25:  SCOPE
1520  006346  032737  002000  001260          BIT     #BIT10,VT5XX    ;IS UNIT EQUIPPED WITH PRINTER.
1521  006354  001002                          BNE     1$              ;YES-TEST IT
1522  006356  000137  006644                  JMP     $EOP            ;NO-EXIT TEST
1523  006362  004537  011442          1$:     JSR     R5,AMSG         ;DISPLAY HEADER
1524  006366  014160                          MPTCNT
1525  006370  004537  011442                  JSR     R5,AMSG         ;ENABLE PRINTER CONTROLLER MODE.
1526  006374  021567                          ENPNTM
1527  006376  012737  000040  001306          MOV     #40,STCHAR      ;LOAD INITIAL CHAR.
1528  006404  013701  001306          2$:     MOV     STCHAR,R1
1529  006410  004537  010170                  JSR     R5,LIC          ;BUILD A 132 COL. LINE.
1530  006414  001316                          PNTWID
1531  006416  012737  000001  010654          MOV     #1,XOFFOK       ;ALLOW XOFF/XON PROTOCOL
1532  006424  004737  010236                  JSR     PC,XPRNT        ;DISPLAY IT.
1533  006430  005237  001306                  INC     STCHAR          ;INCREMENT 1ST CHAR.
1534  006434  023737  001306  001310          CMP     STCHAR,LASTCH   ;ISSUED 92 LINES?
1535  006442  001360                          BNE     2$              ;NO-LOOP
1536  006444  004537  011442                  JSR     R5,AMSG         ;YES-DISABLE PRINTER
1537  006450  021574                          EXPNTM                  ;CONTROLLER MODE.
```

```
1538  006452  004737  011262                  JSR     PC,DELAY        ;TEST DELAY SWITCH AND EXIT.
1539
1540                                  ;;*****************************************************************
  (3)                                 ;*TEST 26        V       PRINT SCREEN TEST
  (3)                                 ;;*****************************************************************
  (2)  006456  000004                 TST26:  SCOPE
1541
1542
1543  006460  004537  011442                  JSR     R5,AMSG         ;DISPLAY PRINT SCREEN MESSAGE
1544  006464  014227                          MPTSCN
1545
1546  006466  004737  013010                  JSR     PC,FILLWC       ;FILL THE SCREEN WITH E
1547
1548  006472  012737  000001  010654          MOV     #1,XOFFOK       ;ALLOW XOFF/XON PROTOCOL
1549  006500  004537  011442                  JSR     R5,AMSG         ;PRINT THEM
1550  006504  017752                          GOPRNT
1551
1552  006506  004737  011262                  JSR     PC,DELAY        ;TEST DELAY SWITCH.
1553
1554                                  ;;*****************************************************************
  (3)                                 ;*TEST 27        W       AUTO PRINT TEST
  (3)                                 ;;*****************************************************************
  (2)  006512  000004                 TST27:  SCOPE
1555
1556  006514  004537  011442                  JSR     R5,AMSG
1557  006520  014264                          M923A                   ;DISPLAY HEADER
1558  006522  004537  011442                  JSR     R5,AMSG         ;ENABLE AUTO-PRINT
1559  006526  020054                          GOAPMD
1560
1561  006530  013737  001266  001314          MOV     VHD,TEMPO       ;LOAD A EXECUTION COUNT
1562  006536  012701  000101                  MOV     #101,R1         ;LOAD A CHARACTER
1563  006542  004737  010134                  JSR     PC,FILBUF       ;LOAD CHARACTER INTO BUFFER
1564  006546  012737  000001  010654  1$:     MOV     #1,XOFFOK
1565  006554  004737  010236                  JSR     PC,XPRNT        ;DISPLAY IT
1566  006560  004737  011262                  JSR     PC,DELAY        ;PROGRAM DELAY
1567  006564  005337  001314                  DEC     TEMPO           ;FINISHED ?
1568  006570  100366                          BPL     1$              ;BR IF NOT
1569  006572  012737  000001  010654          MOV     #1,XOFFOK
1570  006600  004537  011442                  JSR     R5,AMSG         ;DISABLE AUTO-PRINT
1571  006604  020061                          GONAPM
1572  006606  004737  011262                  JSR     PC,DELAY        ;ANOTHER DELAY
1573
1574  006612  000137  006644                  JMP     $EOP            ;END OF PASS
1575  006616  022131                  DISPCH: MTEXT0
1576  006620  022235                          MTEXT1
1577  006622  022336                          MTEXT2
1578  006624  022440                          MTEXT3
1579  006626  022523                          MTEXT4
1580  006630  022625                          MTEXT5
1581  006632  022726                          MTEXT6
1582  006634  022760                          MTEXT7
1583  006636  023060                          MTEXT8
1584  006640  000000                          0
1585  006642  000000                          0
1586
```

```
1607
1608
(1)
(2)                                     ;;*********************************************************************
(1)                                     ;*INCREMENT THE PASS NUMBER ($PASS)
(1)                                     ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
(1)                                     ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
(1)                                     ;*IF THERES A MONITOR GO TO IT
(1)                                     ;*IF THERE ISN'T JUMP TO NOWEOP
(1)
(1)  006644                             $EOP:
(3)  006644  000004                             SCOPE
(3)  006646  005737  001244                     TST     LAST                    ;TEST IF MORE
(3)  006652  001414                             BEQ     1$                      ;BR IF NONE
(3)  006654  023737  001244  001246             CMP     LAST,VTNOW              ;IS THIS THE LAST ONE
(3)  006662  001410                             BEQ     1$                      ;BR IF YES
(3)  006664  062737  000010  001246             ADD     #10,VTNOW
(3)  006672  012737  002466  002402             MOV     #TST2,WHERE
(3)  006700  000137  002014                     JMP     RSTRT                   ;TEST NEXT ONE
(3)  006704  005737  001100             1$:     TST     $PASS                   ;TEST IF FIRST PASS
(3)  006710  001002                             BNE     2$                      ;BR IF NOT
(3)  006712  104401  020300                     TYPE    ,PASHED                 ;TYPE EOP HEADER
(3)  006716  000005                      2$:    RESET
(3)  006720  005737  001332                     TST     WFTEST          ;TEST IF W.F. MODE
(3)  006724  001402                             BEQ     3$              ;BR IF NOT
(3)  006726  004737  011370                     JSR     PC,ADELAY       ;EXTRA DELAY
(3)  006732  000240                      3$:    NOP
(3)
(1)  006734  005037  001102                     CLR     $TSTNM          ;;ZERO THE TEST NUMBER
(1)  006740  005237  001100                     INC     $PASS           ;;INCREMENT THE PASS NUMBER
(1)  006744  042737  100000  001100             BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
(1)  006752  005327                             DEC     (PC)+           ;;LOOP?
(1)  006754  000001                      $EOPCT: .WORD   1
(1)  006756  003022                             BGT     $DOAGN          ;;YES
(1)  006760  012737                             MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
(1)  006762  000001                      $ENDCT: .WORD   1
(1)  006764  006754                             $EOPCT
(1)  006766  104401  007033                     TYPE    ,$ENDMG         ;;TYPE "END PASS #"
(2)  006772  013746  001100                     MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
(2)  006776  104405                             TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
(1)  007000  104401  007030                     TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
(1)  007004  013700  000042              $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
(1)  007010  001405                             BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
(1)  007012  000005                             RESET                   ;;CLEAR THE WORLD
(1)  007014  004710                      $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
(1)  007016  000240                             NOP                     ;;SAVE ROOM
(1)  007020  000240                             NOP                     ;;FOR
(1)  007022  000240                             NOP                     ;;ACT11
(1)  007024                              $DOAGN:
(1)  007024  000137                             JMP     @(PC)+          ;;RETURN
(1)  007026  007050                      $RTNAD: .WORD   NOWEOP
(1)  007030     377      377      000    $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
(1)  007033     015  042412  042116    $ENDMG: .ASCIZ  <15><12>/END PASS #/
(1)  007040  050040  051501  020123
(1)  007046  000043
1609 007050  005737  001274             NOWEOP: TST     PRTCNT
```

```
1610  007054  100002                        BPL     11$
1611  007056  105337  001274                DECB    PRTCNT
1612  007062  012737  002466  002402  11$:  MOV     #TST2,WHERE
1613  007070  013746  001112                MOV     $ERTTL,-(SP)
1614  007074  104405                        TYPDS
1615  007076  104401  007030                TYPE    ,$ENULL
1616  007102  000137  002006                JMP     $BEGIN
1620                                  ;;*****************************************************************
  (3)                                 ;*TEST 30        X       KEYBOARD OCTAL VALUE LOOP
  (3)                                 ;;*****************************************************************
  (2)  007106  000004                 TST30:  SCOPE
1621  007110  012706  001100           KRBECO: MOV     #STACK,SP
1622  007114  004537  011442                   JSR     R5,AMSG                 ;DISPLAY HEADER
1623  007120  016202                           MKE
1624  007122  004737  013052           1$:     JSR     PC,GETCHR               ;GET CHAR
1625  007126  000775                           BR      1$                ;;BR BACK IF NO INPUT
1626  007130  004737  011462                   JSR     PC,OCTAL                ;CONVERT R0 TO OCTAL
1627  007134  113737  011552  016425           MOVB    DIG0,MKEB               ;LOAD DIGIT
1628  007142  113737  011554  016426           MOVB    DIG1,MKEB+1             ;LOAD DIGIT
1629  007150  113737  011556  016427           MOVB    DIG2,MKEB+2             ;LOAD DIGIT
1630  007156  042700  177600                   BIC     #177600,R0
1631  007162  001001                           BNE     10$
1632  007164  005200                           INC     R0
1633  007166  012701  007276           10$:    MOV     #BFCHR,R1               ;LOAD POINTER
1634  007172  121100                 5$:     CMPB    (R1),R0                 ;TEST IF = TO VALUE IN TABLE ?
1635  007174  001403                           BEQ     3$                      ;BR IF FOUND
1636  007176  005721                           TST     (R1)+                   ;MOVE POINTER
1637  007200  001374                           BNE     5$                      ;BR IF MORE
1638  007202  000407                           BR      2$                      ;BR IF NOT IN LIST
1639  007204  062701  000040           3$:     ADD     #BFCHAR-BFCHR,R1        ;UPDATE POINTER
1640  007210  112137  016420                   MOVB    (R1)+,MKEA1             ;LOAD 1ST CHAR
1641  007214  112137  016421                   MOVB    (R1)+,MKEA1+1           ;LOAD 2ND
1642  007220  000420                           BR      4$
1643  007222  120027  000040           2$:     CMPB    R0,#40                  ;TEST IF LESS THAN 40
1644  007226  101010                           BHI     6$                      ;BR IF ABOVE
1645  007230  062700  000100                   ADD     #100,R0                 ;MAKE PRINTABLE
1646  007234  110037  016421                   MOVB    R0,MKEA1+1              ;SAVE CHAR
1647  007240  112737  000136  016420           MOVB    #136,MKEA1              ;ADD A '↑' BEFORE CHARACTER
1648  007246  000405                           BR      4$
1649  007250  112737  000040  016421  6$:     MOVB    #40,MKEA1+1             ;LOAD SPACE
1650  007256  110037  016420                   MOVB    R0,MKEA1                ;LOAD CHARACTER
1651  007262  005237  010666           4$:     INC     IGNORE                  ;IGNORE DOUBLE CHARACTER FLAG
1652  007266  004537  011442                   JSR     R5,AMSG                 ;DISPLAY MESSAGE
1653  007272  016407                           MKEA
1654  007274  000712                           BR      1$                      ;LOOP BACK
1655
1656                                  ;TABLE OF DEFINED CHARACTERS
1657
1658  007276  000007                 BFCHR:  7                               ;BELL CODE
1659  007300  000010                         10                              ;CURSOR LEFT CODE
1660  007302  000011                         11                              ;TAB CODE
1661  007304  000012                         12                              ;LINE FEED CODE
1662  007306  000015                         15                              ;CARRIAGE RETURN CODE
1663  007310  000033                         33                              ;ESCAPE CODE
1664  007312  000040                         40                              ;SPACE CODE
1665  007314  000177                         177                             ;DELETE CODE
```

```
                                    0
                                    0
                                    0
                                    0
                                    0
                                    0
                                    0
                                    0

                        ;DEFINED CHARACTER EQUIL

                        BFCHAR: .ASCII  /BL/            ;BELL
                                .ASCII  /CL/            ;CURSOR LEFT
                                .ASCII  /HT/            ;H TAB
                                .ASCII  /LF/            ;LINE FEED
                                .ASCII  /CR/            ;CARIAGE RETURN
                                .ASCII  /ES/            ;ESCAPE
                                .ASCII  /SP/            ;SPACE
                                .ASCII  /DE/            ;DELETE
                                0,0,0,0,0,0,0,0,0,0

                                .EVEN
                        ;;*******************************************************************
                        ;*TEST 31        Y       KEYBOARD CHARACTER TEST
                        ;*******************************************************************
                        $T31:   SCOPE
                        TST:    BIT     #BIT09,VT5XX    ;IS UNIT A VT52?
                                BEQ     1$
                                MOV     #VS2RW,R3       ;YES-SET UP FOR LOWER CASE CHAR.
                                BR      2$
                        1$:     MOV     #VS0RW,R3       ;NO-SET UP FOR UPPER CASE CHAR.
                        2$:     MOV     #STACK,SP
                        A:      JSR     R5,AMSG         ;DISPLAY HEADER
                                MKB
                                MOV     (R3)+,R2        ;LOAD ROW #
                                BIT     #BIT09,VT5XX    ;UNIT A VT52?
                                BEQ     1$              ;NO-USE UPPER CASE KEYBOARD.
                                JSR     R5,AMSG         ;ISSUE VT52 ROW1 MESSAGE.
                                MKBB2
                                BR      2$
                        1$:     JSR     R5,AMSG
                                MKBB                    ;TOP ROW
                        2$:     JSR     PC,TSTROW       ;CHECK THE ROW

                        B:      MOV     (R3)+,R2        ;LOAD ROW #
                                JSR     R5,AMSG         ;2ND ROW
                                MKBC
                                JSR     PC,TSTROW       ;CHECK 2ND ROW

                        C:      MOV     (R3)+,R2        ;LOAD ROW #
                                BIT     #BIT09,VT5XX    ;UNIT A VT52?
                                BEQ     1$              ;NO-USE UPPER CASE KEYBOARD.
                                JSR     R5,AMSG         ;ISSUE VT52 ROW 3 MESSAGE.
                                MKBD2
```

```
1716  007530  000403                        BR      2$
1717  007532  004537  011442        1$:     JSR     R5,AMSG
1718  007536  014736                        MKBD
1719  007540  004737  011560        2$:     JSR     PC,TSTROW       ;CHECK ROW 3
1720
1721  007544  012302                        MOV     (R3)+,R2        ;LOAD ROW #
1722  007546  004537  011442                JSR     R5,AMSG
1723  007552  015131                        MKBE
1724  007554  004737  011560                JSR     PC,TSTROW       ;CHECK ROW 4
1725
1726  007560  012702  021176                MOV     #ROW5,R2
1727  007564  004537  011442                JSR     R5,AMSG
1728  007570  015251                        MKBF
1729  007572  004737  011560                JSR     PC,TSTROW       ;CHECK ROW 5
1730
1731                          ;TEST THE "LEFT"-SHIFT KEY
1732
1733  007576  004537  011442        D:      JSR     R5,AMSG         ;DEPRESS THE "LEFT-SHIFT" KEY
1734  007602  015307                        MKBG
1735  007604  012302                        MOV     (R3)+,R2        ;LOAD ROW 1 SHIFTED TABLE
1736  007606  032737  001000  001260        BIT     #BIT09,VTSXX    ;UNIT A VT52?
1737  007614  001404                        BEQ     1$              ;NO-USE UPPER CASE KEYBOARD.
1738  007616  004537  011442                JSR     R5,AMSG         ;ISSUE VT52 ROW1 MESSAGE.
1739  007622  014614                        MKBB2
1740  007624  000403                        BR      2$
1741  007626  004537  011442        1$:     JSR     R5,AMSG         ;TEST ROW 1
1742  007632  014477                        MKBB
1743  007634  004737  011560        2$:     JSR     PC,TSTROW       ;TEST THE ROW
1744  007640  004537  011442                JSR     R5,AMSG         ;RELEASE THE SHIFT KEY
1745  007644  014355                        MKB1
1746
1747                          ;TEST THE "RIGHT-SHIFT" KEY
1748
1749  007646  004537  011442        E:      JSR     R5,AMSG         ;SET THE "RIGHT-SHIFT" KEY
1750  007652  015356                        MKBGA
1751  007654  012302                        MOV     (R3)+,R2        ;LOAD TABLE POINTER
1752  007656  032737  001000  001260        BIT     #BIT09,VTSXX    ;UNIT A VT52?
1753  007664  001404                        BEQ     1$              ;NO-USE UPPER CASE KEYBOARD.
1754  007666  004537  011442                JSR     R5,AMSG         ;ISSUE VT52 ROW1 MESSAGE.
1755  007672  014614                        MKBB2
1756  007674  000403                        BR      2$
1757  007676  004537  011442        1$:     JSR     R5,AMSG
1758  007702  014477                        MKBB
1759  007704  004737  011560        2$:     JSR     PC,TSTROW       ;TEST THE ROW AGAIN WITH THE RIGHT-SHIFT SET
1760  007710  004537  011442                JSR     R5,AMSG
1761  007714  014355                        MKB1                    ;RELEASE SKIFT KEY
1762                          ;TEST THE CONTROL MODE
1763
1764  007716  004537  011442        F:      JSR     R5,AMSG         ;SET CTRL
1765  007722  015426                        MKBH
1766  007724  012302                        MOV     (R3)+,R2        ;LOAD ROW 1 CTRL TABLE
1767  007726  032737  001000  001260        BIT     #BIT09,VTSXX    ;UNIT A VT52?
1768  007734  001404                        BEQ     1$              ;NO-USE UPPER CASE KEYBOARD.
1769  007736  004537  011442                JSR     R5,AMSG         ;ISSUE VT52 ROW1 MESSAGE.
1770  007742  014614                        MKBB2
1771  007744  000403                        BR      2$
```

```
1772  007746  004537  011442          1$:   JSR     R5,AMSG
1773  007752  014477                        MKBB
1774  007754  004737  011560          2$:   JSR     PC,TSTROW       ;TEST THE ROW
1775
1776                                  ;KEYPAD TEST FOR VT50H AND VT52
1777
1778  007760  032737  010000  001260        BIT     #BIT12,VT5XX            ;VT50H EXTRA KEYPAD?
1779  007766  001436                        BEQ     KRBDON                  ;NO EXIT TEST
1780  007770  004537  011442                JSR     R5,AMSG                 ;TYPE TEST ID
1781  007774  015700                        MKBN
1782  007776  004537  011442                JSR     R5,AMSG                 ;TYPE INSTRUCTIONS
1783  010002  022046                        INSTR
1784  010004  012702  021602                MOV     #KPTAB,R2               ;SET TABLE POINTER
1785  010010  004737  012164                CALL    TSTKPS                  ;DO THE TEST
1786  010014  032737  001000  001260        BIT     #1000,VT5XX             ;IS UUT VT52?
1787  010022  001420                        BEQ     KRBDON                  ;NO EXIT TEST
1788  010024  004537  011442                JSR     R5,AMSG                 ;TYPE TEST ID
1789  010030  015727                        MKBS2
1790  010032  004537  011442                JSR     R5,AMSG                 ;TYPE INSTRUCTIONS
1791  010036  022046                        INSTR
1792  010040  004537  011442                JSR     R5,AMSG                 ;ENTER ALT-MODE
1793  010044  021543                        ENAKP
1794  010046  012702  021660                MOV     #AKPTAB,R2              ;SET TABLE POINTER
1795  010052  004737  012164                CALL    TSTKPS                  ;DO THE TEST
1796  010056  004537  011442                JSR     R5,AMSG                 ;EXIT ALT-MODE
1797  010062  021550                        EXAKP
1798                                  ;COMPLETION OF KEYBOARD-KEYPAD TEST
1799
1800  010064  004537  011442          KRBDON: JSR     R5,AMSG         ;END OF KEYBOARD TEST
1801  010070  016066                        MKBR
1802  010072  000137  007404                JMP     KRBTST          ;LOOP
1803                                  ;;*****************************************************************
  (3)                             ;*TEST 32       Z       KEYBOARD ECHO LOOP
  (3)                             ;;*****************************************************************
  (2)  010076  000004             TST32:  SCOPE
1804  010100  012706  001100      KRBECH: MOV     #STACK,SP
1805  010104  004537  011442              JSR     R5,AMSG         ;DISPLAY HEADER
1806  010110  016434                      MKEH
1807
1808  010112  004737  013052          1$:   JSR     PC,GETCHR       ;GET CHARACTER
1809  010116  000775                        BR      1$
1810  010120  110077  171160              MOVB    R0,@VTOB        ;LOAD THE CHARACTER
1811  010124  105777  171152          2$:   TSTB    @VTOS           ;WAIT FOR DONE
1812  010130  100375                        BPL     2$
1813  010132  000767                        BR      1$              ;LOOP BACK
1817
1818                                  ;LOAD A SINGLE CHARACTER ACROSS THE SCREEN WIDTH
1819                                  ;
1820
1821  010134  013702  001320      FILBUF: MOV     WIDTH,R2        ;LOAD WIDTH VALUE
1822  010140  012700  025256      FILBFB: MOV     #BUFFER,R0      ;SET-UP BUFFER POINTER
1823  010144  112720  000015              MOVB    #15,(R0)+       ;LOAD 'CR'
1824  010150  112720  000012              MOVB    #12,(R0)+       ;LOAD 'FL'
1825  010154  110120              FILBFA: MOVB    R1,(R0)+        ;SAVE THE CHARACTER IN THE BUFFER
1826  010156  005302                      DEC     R2              ;FINISHED?
1827  010160  001375                      BNE     FILBFA          ;BRANCH IF NOT COMPLETED
```

```
1828  010162  112710  000377              MOVB    #377,(R0)       ;LOAD TERM.
1829  010166  000207                      RTS     PC              ;EXIT
1830
1831                                  ;LOAD A INCREMENTING CHARACTER ACROSS THE SCREEN WIDTH
1832                                  ;ONLY 40 THRU 177 ARE LEGAL CHARACTERS
1833
1834  010170  012700  025256        LIC:    MOV     #BUFFER,R0      ;SET-UP BUFFER POINTER
1835  010174  013502                        MOV     @(5)+,R2        ;SET-UP WIDTH
1836  010176  112720  000015                MOVB    #15.,(R0)+      ;LOAD 'CR'
1837  010202  112720  000012                MOVB    #12.,(R0)+      ;LOAD 'LF'
1838  010206  110120                LICA:   MOVB    R1,(0)+         ;SAVE A CHARACTER IN THE BUFFER
1839  010210  005201                        INC     R1              ;UPDATE THE CHARACTER
1940  010212  023701  001310                CMP     LASTCH,R1       ;TEST FOR
1841  010216  001002                        BNE     LICB            ;BRANCH IF NOT
1842  010220  012701  000040                MOV     #40,R1          ;MAKE A LEGAL CHARACTER
1843  010224  005302                LICB:   DEC     R2              ;DECREMENT COUNT
1844  010226  001367                        BNE     LICA            ;BRANCH IF NOT COMPLETED
1845  010230  112710  000377                MOVB    #377,(R0)       ;LOAD TERM
1846  010234  000205                        RTS     R5              ;EXIT
1847
1848                                  ;DISPLAY SUBROUTINE
1849
1850  010236  012700  025256        XPRNT:  MOV     #BUFFER,R0      ;SETUP BUFFER POINTER
1851  010242  105777  171034        XPRNTA: TSTB    @VTOS           ;TEST READY
1852  010246  100404                        BMI     XPRNTB          ;BRANCH IF SET
1853  010250  005777  171026                TST     @VTOS           ;TEST ERROR
1854  010254  100372                        BPL     XPRNTA          ;BRANCH IF RESET
1855  010256  104001                        ERROR   1               ;ERROR FLAG SET ON TRANSMITTER STATUS
1856  010260  112001                XPRNTB: MOVB    (0)+,R1
1857  010262  100563                        BMI     1$              ;BR IF MINUS
1858  010264  122701  000033        5$:     CMPB    #33,R1          ;TEST FOR ESC
1859  010270  001003                        BNE     3$              ;BR IF NOT
1860  010272  005237  010664                INC     ANESC           ;SET SOFT FLAG
1861  010276  000402                        BR      4$
1862  010300  005037  010664        3$:     CLR     ANESC           ;CLEAR SOFT FLAG
1863  010304  110177  170774        4$:     MOVB    R1,@VTOB        ;LOAD CHAR
1864  010310  105777  170762                TSTB    @VTIS           ;TEST INPUT FLAG
1865  010314  100352                        BPL     XPRNTA          ;BR IF CLEARED
1866  010316  005737  010664                TST     ANESC           ;TEST IF ESC
1867  010322  001347                        BNE     XPRNTA
1868  010324  013737  001254  013124  52$:  MOV     TIME0,TIME1
1869  010332  005037  013126                CLR     TIME2           ;LOAD DELAY
1870  010336  105777  170734        2$:     TSTB    @VTIS           ;TEST IF INPUT FLAG
1871  010342  100407                        BMI     53$             ;BR IF SET
1872  010344  005337  013126                DEC     TIME2           ;DELAY
1873  010350  001372                        BNE     2$
1874  010352  005337  013124                DEC     TIME1           ;DELAY
1875  010356  001367                        BNE     2$
1876  010360  000440                        BR      60$             ;REPORT ERROR
1877  010362  005737  010666        53$:    TST     IGNORE          ;IGNORE KEYBOARD CHARACTERS ?
1878  010366  001104                        BNE     15$             ;BR IF YES
1879  010370  017737  170704  001126        MOV     @VTIB,$BDDAT    ;READ CHAR
1880  010376  042737  177600  001126        BIC     #177600,$BDDAT  ;MASK
1881  010404  022737  000021  001126        CMP     #XON,$BDDAT     ;TEST FOR X ON
1882  010412  001003                        BNE     50$
1883  010414  005237  010662                INC     XONRC           ;SET FLAG
```

```
1884  010420  000710                          BR      XPRNTA          ;START AGAIN
1885  010422  022737  000023  001126  50$:    CMP     #XOFF,$BDDAT    ;TEST FOR X OFF
1886  010430  001020                          BNE     51$             ;BR IF NOT
1887  010432  005237  010660                  INC     XOFFRC
1888  010436  005737  010654                  TST     XOFFOK          ;XOFF ENABLED ?
1889  010442  001730                          BEQ     52$
1890  010444  012737  000001  010656          MOV     #1,AXOFF        ;SET X OFF SOFT FLAG
1891  010452  005737  010652                  TST     XOFFBR          ;TEST
1892  010456  001271                          BNE     XPRNTA          ;BR BACK
1893  010460  000721                          BR      52$             ;BR
1894  010462  012737  000000  001124  60$:    MOV     #0,$GDDAT
1895  010470  000437                          BR      13$
1896  010472  105777  170442          51$:    TSTB    @#SWR           ;TEST SWR
1897  010476  100371                          BPL     60$             ;BR IF CLEARED
1898  010500  012737  000057  001124          MOV     #'/,$GDDAT      ;LOAD EXPECTED
1899  010506  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;COMPARE
1900  010514  001437                          BEQ     14$             ;BR IF EQUAL
1901  010516  012737  000134  001124          MOV     #'\,$GDDAT      ;LOAD EXPECTED
1902  010524  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;COMPARE
1903  010532  001016                          BNE     13$             ;BR IF NOT
1904
1905                                  ;"\" OR LOOP EXIT
1906
1907  010534  012737  000001  010650          MOV     #1,LOOP         ;SET SOFT FLAG
1908  010542  012737  017556  010706          MOV     #MQ0,FINDTA     ;SETUP MESSAGE
1909  010550  005037  010652          16$:    CLR     XOFFBR
1910  010554  005037  010654                  CLR     XOFFOK
1911  010560  005037  010666                  CLR     IGNORE
1912  010564  000137  010676                  JMP     FINDOT
1913
1914  010570  005737  010666          13$:    TST     IGNORE          ;IGNORE INPUT CHARACTER
1915  010574  001001                          BNE     15$
1916  010576  104004                          ERROR   4               ;UNEXPECTED OR INCORRECT INPUT FLAG
1917  010600  000240                  15$:    NOP
1918  010602  000240                          NOP
1919  010604  000240                          NOP
1920  010606  000240                          NOP
1921  010610  000137  010242                  JMP     XPRNTA
1922
1923                                  ;"/" OR STANDARD EXIT
1924
1925  010614  005037  010650          14$:    CLR     LOOP
1926  010620  012737  017643  010706          MOV     #MQ1,FINDTA     ;SETUP MESSAGE
1927  010626  000137  010550                  JMP     16$
1928
1929                                  ;NORMAL EXIT
1930
1931  010632  005037  010654          1$:     CLR     XOFFOK
1932  010636  005037  010666                  CLR     IGNORE
1933  010642  005037  010652                  CLR     XOFFBR
1934  010646  000207                          RTS     PC              ;EXIT
1935
1936  010650  000000                  LOOP:   0
1937  010652  000000                  XOFFBR: 0
1938  010654  000000                  XOFFOK: 0
1939  010656  000000                  AXOFF:  0
```

```
1940  010660  000000                  XOFFRC: 0
1941  010662  000000                  XONRC:  0
1942  010664  000000                  ANESC:  0
1943  010666  000000                  IGNORE: 0                ;WHEN SET IGNORE KEYBOARD FLAGS
1944
1945                                  ;DETERMINE THE  TEST TO GO TO
1946  010670  004537  011442          FNDA:   JSR     R5,AMSG
1947  010674  017731                          MQ2                      ;ERROR ASK AGAIN
1948  010676  012706  001100          FINDOT: MOV     #STACK,SP
1949  010702  004537  011442                  JSR     R5,AMSG
1950  010706  017643                  FINDTA: MQ1
1951  010710  004737  013052                  JSR     PC,GETCHR
1952  010714  000770                          BR      FINDOT
1953  010716  042700  100600                  BIC     #100600,R0       ;MASK
1954  010722  122700  000101                  CMPB    #'A,R0           ;TEST FOR NUMBER
1955  010726  101360                          BHI     FNDA
1956  010730  122700  000132                  CMPB    #'Z,R0           ;TEST FOR OTHERS
1957  010734  103755                          BLO     FNDA
1958  010736  042700  177740                  BIC     #177740,R0       ;MAKE 0-32
1959  010742  005300                          DEC     R0
1960  010744  110037  001102                  MOVB    R0,$TSTNM        ;LOAD THAT TEST #
1961  010750  006300                          ASL     R0
1962  010752  005760  010776                  TST     DSPCH(R0)        ;TEST IF VALID
1963  010756  001744                          BEQ     FNDA                     ;BR IF NOT
1964  010760  000240                          NOP
1965  010762  000240                          NOP
1966  010764  016037  010776  001106          MOV     DSPCH(R0),$LPADR         ;LOAD LOOP ADDRESS
1967  010772  000170  010776                  JMP     @DSPCH(R0)       ;GO TO THAT TEST
1968
1969                                  ;SUBTEST DISPATCH TABLE
1970
1971  010776  002044                  DSPCH:  TST1+2
1972  011000  002470                          TST2+2
1973  011002  002510                          TST3+2
1974  011004  002564                          TST4+2
1975  011006  002664                          TST5+2
1976  011010  002766                          TST6+2
1977  011012  003430                          TST7+2
1978  011014  004000                          TST10+2
1979  011016  004156                          TST11+2
1980  011020  004304                          TST12+2
1981  011022  004342                          TST13+2
1982  011024  005102                          TST14+2
1983  011026  005252                          TST15+2
1984  011030  005364                          TST16+2
1985  011032  005534                          TST17+2
1986  011034  005570                          TST20+2
1987  011036  005660                          TST21+2
1988  011040  006012                          TST22+2
1989  011042  006146                          TST23+2
1990  011044  006270                          TST24+2
1991  011046  006346                          TST25+2
1992  011050  006460                          TST26+2
1993  011052  006514                          TST27+2
1994  011054  007110                          TST30+2
1995  011056  007404                          TST31+2
```

```
                                        ;SUBROUTINE TO LOAD COPIER TEST
                                FIRLST: MOV     #BUFFER,R0
011060  012700  025256                  MOVB    #15,(R0)+       ;LOAD CR
011064  112720  000015                  MOVB    #12,(R0)+       ;LOAD LF
011070  112720  000012                  MOVB    #'E,(R0)+
011074  112720  000105                  MOVB    #'E,(R0)+
011100  112720  000105                  MOVB    #'E,(R0)+
011104  112720  000105                  MOV     WIDTH,R2        ;LOAD WIDTH
011110  013702  001320                  SUB     VH1,R2
011114  163702  001270                  DEC     R2
011120  005302                  1$:     MOVB    #40,(R0)+       ;LOAD A SPACE
011122  112720  000040                  DEC     R2              ;DONE ?
011126  005302                          BPL     1$              ;BR UNTIL DONE
011130  100374                          MOVB    #'E,(R0)+
011132  112720  000105                  MOVB    #'E,(R0)+
011136  112720  000105                  MOVB    #'E,(R0)+       ;LOAD 80 TH
011142  112720  000105                  MOVB    #377,(R0)       ;LOAD TERM
011146  112710  000377                  RTS     PC              ;EXIT
011152  000207

                                        ;SUBROUTINE FOR THE DATA PATH TEST

011154  012700  025256          DTPSR:  MOV     #BUFFER,R0
011160  012501                  DTPSRB: MOV     (5)+,R1         ;GET FIRST CHARACTER
011162  012502                          MOV     (5)+,R2         ;GET SECOND CHARACTER
011164  013703  001320                  MOV     WIDTH,R3                ;SET THE WIDTH
011170  006203                          ASR     R3              ;DIVIDE BY 2
011172  112720  000015                  MOVB    #15,(R0)+       ;LOAD 'CR'
011176  112720  000012                  MOVB    #12,(R0)+       ;LOAD 'LF'
011202  110120                  DTPSRA: MOVB    R1,(0)+         ;
011204  110220                          MOVB    R2,(0)+
011206  005303                          DEC     R3
011210  100374                          BPL     DTPSRA
011212  112710  000377                  MOVB    #377,(R0)       ;LOAD TERM
011216  000205                          RTS     R5

                                        ;SUBROUTINE TO LOAD BUFFER WITH GRAPHICS CHARACTERS

011220  012700  025256          GBBUF:  MOV     #BUFFER,R0      ;LOAD BUFFER ADDRESS
011224  012701  000136                  MOV     #136,R1         ;LOAD INITIAL CHAR.
011230  010402                          MOV     R4,R2           ;LOAD BUFFER COUNT
011232  110120                  1$:     MOVB    R1,(R0)+        ;INSERT A CHAR. IN THE BUFFER
011234  005201                          INC     R1              ;INCREMENT CHAR.
011236  122701  000177                  CMPB    #177,R1         ;AT END OF GRAPHICS STRING?
011242  001002                          BNE     2$              ;NO
011244  012701  000136                  MOV     #136,R1         ;YES-RESET IT TO 1ST GRAPH. CHAR.
011250  005302                  2$:     DEC     R2              ;DECREMENT BUFFER COUNT.
011252  001367                          BNE     1$              ;NOT AT END-LOOP
011254  112710  000377                  MOVB    #377,(R0)       ;END OF BUFFER-INSERT TERMINATOR
011260  000207                          RTS     PC              ;AND EXIT.

                                        ;PROGRAM DELAY ROUTINE

011262  013737  001256  011364  DELAY:  MOV     SUBTST,10$      ;LOAD COUNT
011270  005037  011366                  CLR     11$
011274  005737  001332                  TST     WFTEST          ;TEST IF W.F. MODE
```

```
2052  011300  001413                        BEQ     2$              ;BR IF NOT
2053  011302  006237  011364                ASR     10$             ;CHANGE DELAY TIMER
2054  011306  006237  011364                ASR     10$
2055  011312  006237  011364                ASR     10$
2056  011316  000240                        NOP
2057  011320  000240                        NOP
2058  011322  000240                        NOP
2059  011324  000240                        NOP
2060  011326  000240
2061  011330  032777  010000  167602  2$:   BIT     #BIT12,@SWR     ;TEST SR
2062  011336  001006                        BNE     3$              ;BR IF SET
2063  011340  005337  011366                DEC     11$             ;DELAY
2064  011344  001371                        BNE     2$
2065  011346  005337  011364                DEC     10$
2066  011352  100366                        BPL     2$              ;DELAY
2067  011354  000240                3$:     NOP
2068  011356  000240                        NOP
2069  011360  000240                        NOP
2070  011362  000207                        RTS     PC              ;EXIT
2071
2072  011364  000002                10$:    2
2073  011366  000000                11$:    0
2074
2075  011370  013737  001256  011436  ADELAY: MOV     SUBTST,10$
2076  011376  005037  011440                CLR     11$
2077  011402  006237  011436                ASR     10$
2078  011406  006237  011436                ASR     10$
2079  011412  005337  011440          2$:   DEC     11$
2080  011416  001375                        BNE     2$
2081  011420  005337  011436                DEC     10$
2082  011424  100372                        BPL     2$
2083  011426  000240                        NOP
2084  011430  000240                        NOP
2085  011432  000240                        NOP
2086  011434  000207                        RTS     PC
2087  011436  000000                10$:    0
2088  011440  000000                11$:    0
2089
2090
2091                                  ;HEADER SUBROUTINE FOR VT-50
2092
2093  011442  012537  011452          AMSG:   MOV     (R5)+,10$       ;GET POINTER
2094  011446  004537  013130          1$:     JSR     R5,MTOB         ;MOVE TO BUFFER
2095  011452  000000                10$:    0
2096  011454  004737  010236          11$:    JSR     PC,XPRNT        ;DISPLAY IT
2097  011460  000205                        RTS     R5              ;EXIT
2098
2099
2100                                  ;OCTAL - 3 BIT CONVERSION
2101
2102  011462  010001                OCTAL:  MOV     R0,R1           ;LOAD R1
2103  011464  042701  177770                BIC     #177770,R1      ;MASK
2104  011470  062701  000060                ADD     #60,R1
2105  011474  110137  011556                MOVB    R1,DIG2         ;SAVE LSD
2106  011500  010001                        MOV     R0,R1
2107  011502  006001                        ROR     R1
```

```
2108  011504  006001                        ROR     R1
2109  011506  006001                        ROR     R1
2110  011510  042701  177770                BIC     #177770,R1
2111  011514  062701  000060                ADD     #60,R1
2112  011520  110137  011554                MOVB    R1,DIG1         ;SAVE IT
2113  011524  010001                        MOV     R0,R1
2114  011526  006101                        ROL     R1
2115  011530  006101                        ROL     R1
2116  011532  000301                        SWAB    R1
2117  011534  042701  177770                BIC     #177770,R1
2118  011540  062701  000060                ADD     #60,R1
2119  011544  110137  011552                MOVB    R1,DIG0         ;SAVE MSD
2120  011550  000207                        RTS     PC              ;EXIT
2121
2122  011552  000000              DIG0:     0
2123  011554  000000              DIG1:     0
2124  011556  000000              DIG2:     0
2125
2126                              ;SUBROUTINE FOR THE KEYBOARD CHARACTER TEST
2130  011560  004537  011442      TSTROW: JSR     R5,AMSG         ;DISPLAY HEADER
2131  011564  014410                      MKBA'
2132  011566  004737  013052      1$:     CALL    GETCHR          ;GET A CHARACTER
2133  011572  000775                      BR      1$              ;WAIT FOR INPUT
2134  011574  004737  011772              CALL    PARITY          ;ADJUST CHAR IF PARITY
2135  011600  120037  011762              CMPB    R0,100$         ;CMP EXPCT AND RCVD
2136  011604  001003                      BNE     2$              ;TYPE ERROR INFO
2137  011606  005722                      TST     (R2)+
2138  011610  100366                      BPL     1$              ;LOOP TILL DONE
2139  011612  000207                      RETURN                  ;EXIT TEST
2140
2141                              ;COME HERE IF EXPECTED NOT EQUAL TO RECVD
2142                              ;CONVERT RESULTS TO OCTAL FOR TYPEOUT
2143
2144  011614  010037  001126      2$:     MOV     R0,$BDDAT       ;LOAD BAD CHARACTER
2145  011620  004737  011462              JSR     PC,OCTAL        ;CONVERT TO OCTAL
2146  011624  113737  011552  016057      MOVB    DIG0,MKBQB      ;LOAD OCTAL #
2147  011632  113737  011554  016060      MOVB    DIG1,MKBQB+1
2148  011640  113737  011556  016061      MOVB    DIG2,MKBQB+2
2149  011646  042700  177600              BIC     #177600,R0
2150  011652  120027  000040              CMPB    R0,#40          ;TEST IF PRINTABLE
2151  011656  101002                      BHI     10$             ;BR IF PRINTABLE
2152  011660  112700  000056              MOVB    #56,R0          ;CONVERT TO A "*" CHARACTER
2153  011664  110037  016053      10$:    MOVB    R0,MKBQ2        ;SAVE CHAR
2154  011670  011200                      MOV     (R2),R0         ;GET GOOD CHAR
2155  011672  053700  011770              BIS     MASK2,R0
2156  011676  010037  001124              MOV     R0,$GDDAT       ;LOAD GOOD CHARACTER
2157  011702  004737  011462              JSR     PC,OCTAL        ;CONVERT IT
2158  011706  113737  011552  016040      MOVB    DIG0,MKBQA      ;LOAD DIGIT
2159  011714  113737  011554  016041      MOVB    DIG1,MKBQA+1
2160  011722  113737  011556  016042      MOVB    DIG2,MKBQA+2
2161  011730  042700  177600              BIC     #177600,R0
2162  011734  110037  016034              MOVB    R0,MKBQ1        ;SAVE CHAR
2163  011740  023737  001276  001144      CMP     VTIS,$TKS       ;TEST IF ON CTY
2164  011746  001403                      BEQ     3$              ;BR IF YES
2165  011750  004537  011442              JSR     R5,AMSG         ;DISPLAY ERROR  MESSAGE
2166  011754  015773                      MKBQ
```

```
2167  011756  104004                  3$:     ERROR   4               ;CHARACTER RECVD NOT EQUAL TO EXPECTED
2168  011760  000702                          BR      1$              ;BR BACK AND TEST THE CHARACTER AGAIN
2169
2170  011762  000000                  100$:   0
2171  011764  000000                  101$:   0
2172  011766  177600                  MASK1:  177600
2173  011770  000000                  MASK2:  0
2174
2175                                  ;TEST TO ADD PARITY TO EXPCTD CHAR
2176  011772  012737  177600  011766  PARITY: MOV     #177600,MASK1
2177  012000  005037  011770                  CLR     MASK2
2178  012004  032777  000004  167126          BIT     #BIT2,@SWR      ;TEST SWR
2179  012012  001416                          BEQ     4$              ;DO NOT TEST PARITY BIT
2180  012014  042737  000200  011766          BIC     #BIT7,MASK1     ;ENABLE PARITY BIT
2181  012022  032777  000002  167110          BIT     #BIT1,@SWR      ;TEST IF FORCED PARITY
2182  012030  001417                          BEQ     5$              ;BR IF NOT FORCED PARITY BIT
2183  012032  032777  000001  167100          BIT     #BIT0,@SWR      ;TEST FOR EVEN/ODD PARITY
2184  012040  001403                          BEQ     4$              ;BR IF ALLWAYS OFF
2185  012042  052737  000200  011770          BIS     #BIT7,MASK2     ;SET BIT 7
2186  012050  111237  011762          4$:     MOVB    (R2),100$       ;GET EXPECTED
2187  012054  053737  011770  011762          BIS     MASK2,100$      ;SET BIT 7 IF EXPECTED
2188  012062  043700  011766                  BIC     @MASK1,R0       ;MASK VALUE READ
2189  012066  000207                          RETURN                  ;EXIT ROUTINE
2190  012070  005037  011764          5$:     CLR     101$            ;CLEAR TEMP
2191  012074  111237  011762                  MOVB    (R2),100$       ;CLEAR CHAR SAVE
2192  012100  006037  011762          20$:    ROR     100$            ;ROTATE CHAR
2193  012104  103002                          BCC     21$             ;BR IF NO CARRY
2194  012106  005237  011764                  INC     101$            ;UPDATE CNT
2195  012112  105737  011762          21$:    TSTB    100$            ;DONE ?
2196  012116  001370                          BNE     20$             ;BR IF NOT
2197  012120  032777  000001  167012          BIT     #BIT0,@SWR      ;TEST EVEN/ODD
2198  012126  001407                          BEQ     23$             ;BR IF OPER. SAYS EVEN
2199  012130  006037  011764                  ROR     101$            ;
2200  012134  103403                          BCS     22$             ;BR IF ODD ALREADY
2201  012136  052737  000200  011770          BIS     #BIT7,MASK2     ;SET PARITY BIT
2202  012144  000741                  22$:    BR      4$              ;BR TO TEST CHAR
2203  012146  006037  011764          23$:    ROR     101$            ;
2204  012152  103003                          BCC     24$             ;BR IF EVEN ALREADY
2205  012154  052737  000200  011770          BIS     #BIT7,MASK2
2206  012162  000732                  24$:    BR      4$              ;BR TO TEST CHAR
2207
2208                                  ;ROUTINE TO TEST THE KEYPAD INPUTS
2209  012164                          ;TSTKPS:
  (2) 012164  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (2) 012166  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (2) 012170  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
  (2) 012172  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2210  012174  012701  025234          71$:    MOV     #GOOD,R1        ;SETUP POINTERS TO DATA
2211  012200  012703  025214                  MOV     #HOLD,R3
2212  012204  010237  025254                  MOV     R2,RETRY
2213  012210  004737  013052          72$:    CALL    GETCHR          ;GET INPUT
2214  012214  000775                          BR      72$             ;LOOP IF NONE
2215  012216  004737  011772                  CALL    PARITY          ;ADD PARITY IF NECESSARY
2216  012222  110023                          MOVB    R0,(R3)+                ;PUT RCVD IN HOLD
2217  012224  113721  011762                  MOVB    100$,(R1)+      ;PUT EXPCTD IN GOOD
2218  012230  105722                          TSTB    (R2)+           ;BUMP POINTER
```

```
                                        CMPB    (R2),#377       ;DONE WITH KEY YET
                                        BNE     72$             ;GET REST OF KEY
                                        MOVB    (R2),(R3)       ;PUT ON DELIMITER
                                        MOVB    (R2)+,(R1)
                                        CALL    TSTDAT          ;COMPARE THE KEY VALUES
                                        TSTB    (R2)            ;END OF TABLE YET
                                        BEQ     73$             ;EXIT TEST
                                        BR      71$             ;NEXT KEY
                                73$:
                                        MOV     (SP)+,R5        ;;POP STACK INTO R5
                                        MOV     (SP)+,R4        ;;POP STACK INTO R4
                                        MOV     (SP)+,R3        ;;POP STACK INTO R3
                                        MOV     (SP)+,R1        ;;POP STACK INTO R1
                                        RETURN

                                ;ROUTINE TO COMPARE EXPECTED AND RECEIVED KEY VALUES
                                ;FOR THE KEYPAD TESTS
                                TSTDAT: MOV     #HOLD,R3        ;SET DATA POINTERS
                                        MOV     #GOOD,R1
                                        MOV     R3,R5
                                        MOV     R1,R4
                                75$:    CMPB    (R1),#377       ;DELIMITER YET
                                        BEQ     74$             ;EXIT ROUTINE
                                        CMPB    (R1)+,(R3)+     ;COMPR EXPCTD AND RCVD
                                        BEQ     75$
                                        MOV     RETRY,R2        ;WE HAVE AN ERROR SOMEWHERE
                                        INC     IGNORE
                                        JSR     R5,AMSG         ;TYPE ERROR INFPO
                                        ERCHR
                                        INC     IGNORE
                                        JSR     R5,AMSG
                                        SHBE
                                        CALL    OUTPUT
                                        INC     IGNORE
                                        JSR     R5,AMSG
                                        AWAS
                                        MOV     R5,R4
                                        CALL    OUTPUT
                                        TSTB    @VTIB           ;CLEAR ANY UNPREDICTED GARBAGE
                                74$:    RETURN

                                ;ROUTINE TO OUTPUT GOOD AND BAD KEYPAD OCTAL VALUES ON SCREEN
                                OUTPUT:
                                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
                                        CLR     R1
                                77$:    MOVB    #40,OUT(R1)     ;FILL AREA WITH SPACES
                                        INC     R1
                                        CMPB    R1,#13          ;FULL YET
                                        BNE     77$
                                        MOV     (SP)+,R1        ;;POP STACK INTO R1
                                        MOVB    (R4)+,R0        ;SETUP KEY VALUE TO CONVERT
                                        CALL    OCTAL           ;DO IT
                                        MOVB    DIG0,OUT
                                        MOVB    DIG1,OUT+1
                                        MOVB    DIG2,OUT+2
                                        CMPB    (R4),#377       ;DELIMITER
```

```
                                        BEQ     76$                     ;EXIT
                                        MOVB    (R4)+,R0                ;SETUP 2ND VALUE OF KEY
                                        CALL    OCTAL                   ;GO IT
                                        MOVB    DIG0,OUT+4
                                        MOVB    DIG1,OUT+5
                                        MOVB    DIG2,OUT+6
                                        CMPB    (R4),#377               ;DELIMITER ?
                                        BEQ     76$                     ;EXIT
                                        MOVB    (R4)+,R0                ;SETUP 3RD VALUE OF KEY
                                        CALL    OCTAL
                                        MOVB    DIG0,OUT+10
                                        MOVB    DIG1,OUT+11
                                        MOVB    DIG2,OUT+12
                                76$:    INC     IGNORE
                                        JSR     R5,AMSG                 ;OUTPUT VALUES ON SCREEN
                                        OUT
                                        RETURN


                                .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                                ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                                ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                                ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                                ;*REPLACED WITH SPACES.
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)               ;;PUT THE BINARY NUMBER ON THE STACK
                                ;*      TYPDS                           ;;GO TO THE ROUTINE
                                ;
012564                          $TYPDS:
012564  010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
012566  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
012570  010246                          MOV     R2,-(SP)                ;;PUSH R2 ON STACK
012572  010346                          MOV     R3,-(SP)                ;;PUSH R3 ON STACK
012574  010546                          MOV     R5,-(SP)                ;;PUSH R5 ON STACK
012576  012746  020200                  MOV     #20200,-(SP)            ;;SET BLANK SWITCH AND SIGN
012602  016605  000020                  MOV     20(SP),R5               ;;GET THE INPUT NUMBER
012606  100004                          BPL     1$                      ;;BR IF INPUT IS POS.
012610  005405                          NEG     R5                      ;;MAKE THE BINARY NUMBER POS.
012612  112766  000055  000001          MOVB    #'-,1(SP)               ;;MAKE THE ASCII NUMBER NEG.
012620  005000                  1$:     CLR     R0                      ;;ZERO THE CONSTANTS INDEX
012622  012703  013000                  MOV     #$DBLK,R3               ;;SETUP THE OUTPUT POINTER
012626  112723  000040                  MOVB    #' ,(R3)+               ;;SET THE FIRST CHARACTER TO A BLANK
012632  005002                  2$:     CLR     R2                      ;;CLEAR THE BCD NUMBER
012634  016001  012770                  MOV     $DTBL(R0),R1            ;;GET THE CONSTANT
012640  160105                  3$:     SUB     R1,R5                   ;;FORM THIS BCD DIGIT
012642  002402                          BLT     4$                      ;;BR IF DONE
012644  005202                          INC     R2                      ;;INCREASE THE BCD DIGIT BY 1
012646  000774                          BR      3$
012650  060105                  4$:     ADD     R1,R5                   ;;ADD BACK THE CONSTANT
012652  005702                          TST     R2                      ;;CHECK IF BCD DIGIT=0
012654  001002                          BNE     5$                      ;;FALL THROUGH IF 0
012656  105716                          TSTB    (SP)                    ;;STILL DOING LEADING 0'S?
012660  100407                          BMI     7$                      ;;BR IF YES
```

```
(1)  012662  106316                  5$:     ASLB    (SP)            ;;MSD?
(1)  012664  103003                          BCC     6$              ;;BR IF NO
(1)  012666  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
(1)  012674  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
(1)  012700  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1)  012704  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
(1)  012706  005720                          TST     (R0)+           ;;JUST INCREMENTING
(1)  012710  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
(1)  012714  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
(1)  012716  003002                          BGT     8$              ;;GO TO EXIT
(1)  012720  010502                          MOV     R5,R2           ;;GET THE LSD
(1)  012722  000764                          BR      6$              ;;GO CHANGE TO ASCII
(1)  012724  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
(1)  012726  100003                          BPL     9$              ;;BR IF NO
(1)  012730  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
(1)  012736  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
(3)  012740  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  012742  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  012744  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  012746  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  012750  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  012752  104401  013000                  TYPE    $DBLK           ;;NOW TYPE THE NUMBER
(1)  012756  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
(1)  012764  012616                          MOV     (SP)+,(SP)
(1)  012766  000002                          RTI                     ;;RETURN TO USER
(1)  012770  023420                  $DTBL:  10000.
(1)  012772  001750                          1000.
(1)  012774  000144                          100.
(1)  012776  000012                          10.
(1)  013000  000004                  $DBLK:  .BLKW   4

2293
2294                                 ;SUBROUTINE TO FILL THE SCREEN WITH AN CHARACTER
2295
2296  013010  012701  000105        FILLWC: MOV     #'E,R1          ;LOAD CHARACTER BYTE
2297  013014  004737  010134                JSR     PC,FILBUF       ;LOAD THE LINE WITH CHAR
2298  013020  013737  001266  013050        MOV     VHO,10$         ;LOAD COUNT
2299  013026  012737  000001  010654  1$:   MOV     #1,XOFFOK       ;INSURE XOFF/XON CONTROL.
2300  013034  004737  010236                JSR     PC,XPRNT        ;DISPLAY THE LINE
2301  013040  005337  013050                DEC     10$
2302  013044  001370                        BNE     1$              ;LOOP UNTIL DONE
2303  013046  000207                        RTS     PC              ;EXIT
2304  013050  000000                10$:    0
2305
2306                                 ;SUBROUTINE TO GET A CHARACTER FROM THE VT50
2307
2308  013052  013737  001254  013124  GETCHR: MOV     TIME0,TIME1     ;LOAD TIME COUNTER
2309  013060  005037  013126                CLR     TIME2
2310
2311  013064  105777  166206          1$:   TSTB    @VTIS           ;TEST INPUT STATUS
2312  013070  100005                        BPL     2$              ;BR IF CLEARED
2313  013072  017700  166202                MOV     @VTIB,R0        ;READ A CHAR
2314  013076  062716  000002                ADD     #2,(SP)         ;UPDATE RETURN
2315  013102  000207                        RTS     PC              ;EXIT
2316
2317  013104  005337  013126          2$:   DEC     TIME2           ;DELAY
2318  013110  001365                        BNE     1$
```

```
2319  013112  005337  013124          DEC     TIME1           ;FINISHED ?
2320  013116  100362                  BPL     1$              ;LOOP TILL TIME EXPIRED
2321  013120  104002                  ERROR   2               ;NO INPUT FLAG FROM DEVICE
2322  013122  000207                  RTS     PC              ;EXIT
2323
2324  013124  000000          TIME1:  0
2325  013126  000000          TIME2:  0
2326
2327                          ;MOVE TO THE OUTPUT BUFFER
2328
2329  013130  012500          MTOB:   MOV     (R5)+,R0        ;LOAD DEST.
2330  013132  012701  025256          MOV     #BUFFER,R1      ;LOAD R1
2331  013136  112021          1$:     MOVB    (R0)+,(R1)+     ;LOAD BYTE
2332  013140  100376                  BPL     1$              ;BR UNTIL DONE
2333  013142  000205                  RTS     R5              ;EXIT
2334
```

```
                                        ;ASCII MESSAGES
013144  006415  046412  044501  TITLE:  .ASCIZ  <15><15><12>\MAINDEC-11-DZVTC-E VT50A, B, VT50H AND VT52 ACCEPTANCE TEST
013245  000115  020011  043040  M91:    .ASCIZ  <15><12>/  FULL SCREEN OF THE CHARACTER E/<15><12><377>
013313  000115  020011  042040  M92:    .ASCIZ  <15><12>/  DATA PATH TEST /<15><12><377>
013342  005015  000040  044523  M93:    .ASCIZ  <15><12>/  SINGLE CHARACTER PER LINE /<15><12><377>
013404  005015  000040  047524  M94:    .ASCIZ  <15><12>/  ROTATING PATTERN /<15><12><377>
013435  000115  020011  041440  M96:    .ASCIZ  <15><12>/  CURSOR MOTION TEST /<15><12><377>
013470  005015  000040  040524  M97:    .ASCIZ  <15><12>/  TAB, BACKSPACE AND BELL TEST  /<15><12><377>
013536  005015  000040  051105  M910:   .ASCIZ  <15><12>/  ERASE LINE TEST  /<15><12><377>
013567  000115  020012  042440  M911:   .ASCIZ  <15><12>/  ERASE SCREEN TEST  /<15><12><377>
013622  005015  020040  044526  M912:   .ASCIZ  <15><12>/  VIDEO COUPLING TEST /<15><12><377>
013656  044033  045033  005015  M914:   .ASCIZ  <33><110><33><112><15><12>/  TERMINAL IDENTIFIER TEST /<15><12><377>
013723  000115  020012  041440  M920:   .ASCIZ  <15><12>/  COPIER PERIMETER/<15><12><377>
013753  000115  020012  042040  M921:   .ASCIZ  <15><12>/  DISCLAIMER STATEMENT/<15><12><377>
014007  000115  020012  044040  M922:   .ASCIZ  <15><12>/  HOLD SCREEN MODE TEST/<15><12><377>
014044  005015  020040  051107  M922A:  .ASCIZ  <15><12>/  GRAPHICS MODE AND REV. LINE FEED TEST/<15><12><377>
014121  000115  015537  015510  M923:   .ASCIZ  <33><137><33><110><33><112>/  AUTO COPY MODE TEST/<15><12><377>
014160  044033  044033  051120  MPTCNT: .ASCIZ  <33><110><33><112>/PRINTER CONTROLLER MODE ENTERED/<15><12><377>
014227  000033  015510  050112  MPTSCN: .ASCIZ  <33><110><33><112>/PRINT A SCREEN OF E'S/<15><12><377>
014264  015510  020112  040440  M923A:  .ASCIZ  <110><33><112>/  AUTO PRINT MODE TEST/<15><12><377>
014321  000115  020012  042513  MKB:    .ASCIZ  <15><12>/ KEYBOARD CHARACTER TEST/<15><12>
014355  012200  046105  040505  MKB1:   .ASCIZ  /RELEASE THE "SHIFT" KEY/<15><12><377>
014410  005015  042514  052106  MKBA:   .ASCIZ  <15><12>/LEFT TO RIGHT IN A ROW, DEPRESS ONE KEY AT A TIME/<15><12><377>
014477  000115  051412  040524  MKBB:   .ASCIZ  <15><12>/STARTING WITH THE TOP ROW EXCEPT THE THIRD/
                                        .ASCIZ  / FROM RIGHT END AND LAST KEYS/<15><12><377>
014614  005015  052123  051101  MKBB2:  .ASCIZ  <15><12>/STARTING WITH THE TOP ROW EXCEPT THE LAST KEY/<15><12><377>
014677  000115  051412  040524  MKBC:   .ASCIZ  <15><12>/START WITH THE SECOND ROW/<15><12><377>
014736  005015  052123  051101  MKBD:   .ASCIZ  <15><12>/START WITH THE THIRD ROW EXCEPT THE CTRL/
015010  005015  040440  042116          .ASCIZ  <15><12>/ AND THE "BLANK" KEYS/<15><12><377>
015043  000115  051412  040524  MKBD2:  .ASCIZ  <15><12>/START WITH THE THIRD ROW ,BEGIN ROW WITH "A" KEY/<15><12><377>
015131  000115  051412  040524  MKBE:   .ASCIZ  <15><12>/START WITH THE FOURTH ROW EXCEPT THE SCROLL/
015206  005015  051454  044510          .ASCIZ  <15><12>/ SHIFT, REPEAT AND AUTO-PRINT/<15><12><377>
015251  000115  051412  040524  MKBF:   .ASCIZ  <15><12>/START WITH THE FIFTH ROW/<15><12><377>
015307  000115  047012  053517  MKBG:   .ASCIZ  <15><12>/NOW HOLD DOWN THE "LEFT-SHIFT" KEY/<15><12><377>
015356  005015  047516  020127  MKBGA:  .ASCIZ  <15><12>/NOW HOLD DOWN THE "RIGHT-SHIFT" KEY/<15><12><377>
015426  005015  047516  020127  MKBH:   .ASCIZ  <15><12>/NOW HOLD DOWN THE "CTRL" KEY/<15><12><377>
015467  000115  051412  040524  MKBI:   .ASCIZ  <15><12>/START WITH THE TOP ROW/<15><12><377>
015522  005015  052123  051101  MKBJ:   .ASCIZ  <15><12>/START WITH THE 2ND ROW/<15><12><377>
015555  000115  051412  040524  MKBK:   .ASCIZ  <15><12>/START  WITH THE 3RD ROW/<15><12><377>
015611  000115  051412  040524  MKBL:   .ASCIZ  <15><12>/START WITH THE 4TH ROW/<15><12><377>
015644  005015  052123  051101  MKBM:   .ASCIZ  <15><12>/START WITH THE LAST ROW/<15><12><377>
015700  005015  053012  032524  MKBN:   .ASCIZ  <15><12><12>/VT50H KEYPAD TEST/<15><12><377>
015727  000115  053012  032524  MKBS2:  .ASCIZ  <15><12>/VT52 ALTERNATE KEYPAD MODE TEST/<15><12><377>
015773  000115  041412  040510  MKBQ:   .ASCIZ  <15><12>/CHARACTER WAS IN ERROR/<15><12>
016025  000107  047517  020104          .ASCII  /GOOD = /
016034  000040  000040  000075  MKBQ1:  .BYTE   40,40,75,40
016040  000040  000040  000040  MKBQA:  .BYTE   40,40,40,40,40
016045  000102  042101  036440          .ASCII  /BAD = /
016053  000040  000040  000075  MKBQ2:  .BYTE   40,40,75,40
016057  000040  000040  000040  MKBQB:  .BYTE   40,40,40,40,15,12,377,0
016066  005015  042513  041131  MKBR:   .ASCIZ  <15><12>/KEYBOARD CHARACTER TEST COMPLETE/<15><12><377>
016134  005015  047111  040526  INVLID: .ASCIZ  <15><12>/INVALID BUSS ADDRESS-PLEASE RETRY/<15><12>

016202  005015  045440  054505  MKE:    .ASCII  <15><12>/ KEYBOARD ASCII AND OCTAL LOOP/<15><12>
```

```
                                .BYTE   15,12
                                .ASCII  /WHEN A KEY IS DEPRESSED, THE ASCII CHARACTER AND/
                                .BYTE   15,12
                                .ASCIZ  / THE THREE DIGIT OCTAL CODE WILL BE ECHOED/
                                .BYTE   15,12,377,0
MKEA:                           .ASCII  <15><12>/CHAR = /
MKEA1:                          .BYTE   40,40,40,75,40
MKEB:                           .BYTE   40,40,40,15,12,377,0
MKEH:                           .ASCIZ  <15><12>/KEYBOARD ECHO LOOP/<15><12>
CRLFB:                          .BYTE   15,12,12,12,12,12,12,12,12,12
CRLFA:                          .BYTE   12,12,12,12,12,377,0
                        ;HOME AND ERASE SCREEN OPCODE
HOMERS:                         .BYTE   33,110,33,112,377,0
PATH:                           .ASCII  <15><12>/T ▒▒▒
```

```
                                DT1:    $ERRPC,VTNOW,TSTNUM,0
                                DT3:    $ERRPC,VTNOW,SAVE4,SAVE2,SAVE3,0
                                DT4:    $ERRPC,VTNOW,TSTNUM,$GDDAT,$BDDAT,0

                                        .SBTTL  KEYBOARD CHARACTER CODE TABLES

                                ;THE ACTUAL KEYBOARD LAYOUT IS REQUIRED

020774  021030  021064  021122  V50RW:  ROW1,ROW2,ROW3,ROW4,ROW1S,ROW1S,ROW1C
021012  021200  021236  021274  V52RW:  ROW12,ROW22,ROW32,ROW42,ROW12S,ROW12S,ROW12C

021030  000033  000061  000062  ROW1:   .WORD   33,61,62,63,64,65,66,67,70,71,60,55,75,100010
021064  000011  000121  000127  ROW2:   .WORD   11,121,127,105,122,124,131,125,111,117,120,133,134,12,100177
021122  000101  000123  000104  ROW3:   .WORD   101,123,104,106,107,110,112,113,114,73,47,100015
021152  000132  000130  000103  ROW4:   .WORD   132,130,103,126,102,116,115,54,56,100057
021176  100040                  ROW5:   .WORD   100040
                                ;VT52 KEYBOARD EQUIVALENCES(LOWER CASE CHAR.)

021200  000033  000061  000062  ROW12:  .WORD   33,61,62,63,64,65,66,67,70,71,60,55,75,140,100010
021236  000011  000161  000167  ROW22:  .WORD   11,161,167,145,162,164,171,165,151,157,160,133,134,12,100177
021274  000141  000163  000144  ROW32:  .WORD   141,163,144,146,147,150,152,153,154,73,47,173,100015
021326  000172  000170  000143  ROW42:  .WORD   172,170,143,166,142,156,155,54,56,100057

                                ;SHIFTED ROW CODES

021352  000033  000041  000100  ROW1S:  .WORD   33,41,100,43,44,45,136,46,52,50,51,137,53,100010
021406  000033  000041  000100  ROW12S: .WORD   33,41,100,43,44,45,136,46,52,50,51,137,53,176,100010

                                ;CONTROL ROW CODES

021444  000033  000021  000022  ROW1C:  .WORD   33,21,22,23,24,25,26,27,30,31,20,15,35,100010
021500  000033  000021  000022  ROW12C: .WORD   33,21,22,23,24,25,26,27,30,31,20,15,35,0,100010
                                        ;VT52   ESCAPE SEQUENCES

021536      033     111     015 REVLF:  .BYTE   33,111,015,377,0        ;REVERSE LINE FEED.

021543      033     075     377 ENAKP:  .BYTE   33,075,377,0,0         ;ENABLE ALTERNATE KEYPAD MODE.
021550      033     076     377 EXAKP:  .BYTE   33,076,377,0,0         ;EXIT ALTERNATE KEYPAD MODE

021555      033     106     000 ENGRAF: .BYTE   33,106,0,377,0         ;ENTER GRAPHICS MODE.
021562      033     107     000 EXGRAF: .BYTE   33,107,0,377,0         ;EXIT GRAPHICS MODE.

021567      033     127     377 ENPNTM: .BYTE   33,127,377,0,0         ;ENABLE PRINTER CONTROLLER MODE.
021574      033     130     377 EXPNTM: .BYTE   33,130,377,0,0         ;DISABLE PRINTER CONTROLLER MODE.

                                ;VT50H KEYPAD CODES
021602      033     120     377 KPTAB:  .BYTE   33,120,-1,33,121,-1,33,122,-1,33,101,-1
021616      067     377     070         .BYTE   67,-1,70,-1,71,-1,33,102,-1
021627      064     377     065         .BYTE   64,-1,65,-1,66,-1,33,103,-1
021640      061     377     062         .BYTE   61,-1,62,-1,63,-1,33,104,-1
021651      060     377     056         .BYTE   60,-1,56,-1,15,-1,0
021660      033     120     377 AKPTAB: .BYTE   33,120,-1,33,121,-1,33,122,-1,33,101,-1
021674      033     077     167         .BYTE   33,77,167,-1,33,77,170,-1,33,77,171,-1,33,102,-1
021713      033     077     164         .BYTE   33,77,164,-1,33,77,165,-1,33,77,166,-1,33,103,-1
```

```
                                .BYTE   33,77,161,-1,33,77,162,-1,33,77,163,-1,33,104,-1
                                .BYTE   33,77,160,-1,33,77,156,-1,33,77,115,-1,0
                        SHBE:   .ASCII  <15><12>/GOOD= /<377>
                        OUT:    .ASCII  /       /<377>
                        AWAS:   .ASCII  <15><12>/BAD=  /<377>
                        ERCHR:  .ASCII  <15><12>/KEY CODE ERROR/<377>
                        INSTR:  .ASCII  /START 1ST ROW LEFT TO RIGHT/<15><12>
                                .ASCII  /PROCEED TO LAST ROW/<15><12><377>

                        MTEXT0: .ASCIZ  <15><12><12>/THIS SOFTWARE IS FURNISHED TO PURCHASER UNDER A LICENSE FOR
                        MTEXT1: .ASCIZ  <15><12>/ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION/<
                        MTEXT2: .ASCIZ  <15><12>/OF DEC'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT/
                        MTEXT3: .ASCIZ  <15><12>/AS MAY OTHERWISE BE PROVIDED IN WRITING BY DEC./<377>

                        MTEXT4: .ASCIZ  <15><12><12>/THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHO
                        MTEXT5: .ASCIZ  <15><12>/NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL//
                        MTEXT6: .ASCIZ  <15><12>/EQUIPMENT CORPORATION./<377>
                        MTEXT7: .ASCIZ  <15><12><12>/DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF
                        MTEXT8: .ASCII  <15><12>/ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC./
                                .BYTE   15,12,377,0
                        .SBTTL  TTY INPUT ROUTINE

                        ;;**********************************************************************
                        .ENABL  LSB

                        .DSABL  LSB


                        ;;**********************************************************************
                        ;;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                        ;;*CALL:
                        ;;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
                        ;;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                        ;;*                              ;;WITH PARITY BIT STRIPPED OFF
                        ;;
                        $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
                                MOV     4(SP),2(SP)     ;;SAVE THE PS
                        1$:     TSTB    @$TKS           ;;WAIT FOR
                                BPL     1$              ;;A CHARACTER
                                MOVB    @$TKB,4(SP)     ;;READ THE TTY
                                BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
                                CMP     4(SP),#23       ;;IS IT A CONTROL-S?
                                BNE     3$              ;;BRANCH IF NO
                        2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
                                BPL     2$              ;;LOOP UNTIL ITS THERE
                                MOVB    @$TKB,-(SP)     ;;GET CHARACTER
                                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
                                CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
                                BNE     2$              ;;IF NOT DISCARD IT
                                BR      1$              ;;YES, RESUME
                        3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
                                BLT     4$              ;;BRANCH IF YES
                                CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
                                BGT     4$              ;;BRANCH IF YES
                                BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
```

```
(1)  023274  000002                    4$:     RTI                             ;;GO BACK TO USER
(2)                                    ;;**************************************************************
(1)                                    ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
(1)                                    ;*CALL:
(1)                                    ;*      RDLIN                           ;;INPUT A STRING FROM THE TTY
(1)                                    ;*      RETURN HERE                     ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
(1)                                    ;*                                      ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
(1)
(1)  023276  010346                    $RDLIN: MOV     R3,-(SP)                ;;SAVE R3
(1)  023300  012703  023404            1$:     MOV     #$TTYIN,R3              ;;GET ADDRESS
(1)  023304  022703  023414            2$:     CMP     #$TTYIN+8.,R3           ;;BUFFER FULL?
(1)  023310  101405                            BLOS    4$                      ;;BR IF YES
(1)  023312  104406                            RDCHR                           ;;GO READ ONE CHARACTER FROM THE TTY
(1)  023314  112613                            MOVB    (SP)+,(R3)              ;;GET CHARACTER
(1)  023316  122713  000177            10$:    CMPB    #177,(R3)               ;;IS IT A RUBOUT
(1)  023322  001003                            BNE     3$                      ;;SKIP IF NOT
(1)  023324  104401  001166            4$:     TYPE    ,$QUES                  ;;TYPE A '?'
(1)  023330  000763                            BR      1$                      ;;CLEAR THE BUFFER AND LOOP
(1)  023332  111337  023402            3$:     MOVB    (R3),9$                 ;;ECHO THE CHARACTER
(1)  023336  104401  023402                    TYPE    ,9$
(1)  023342  122723  000015                    CMPB    #15,(R3)+               ;;CHECK FOR RETURN
(1)  023346  001356                            BNE     2$                      ;;LOOP IF NOT RETURN
(1)  023350  105063  177777                    CLRB    -1(R3)                  ;;CLEAR RETURN (THE 15)
(1)  023354  104401  001170                    TYPE    ,$LF                    ;;TYPE A LINE FEED
(1)  023360  012603                            MOV     (SP)+,R3                ;;RESTORE R3
(1)  023362  011646                            MOV     (SP),-(SP)              ;;ADJUST THE STACK AND PUT ADDRESS OF THE
(1)  023364  016666  000004  000002            MOV     4(SP),2(SP)             ;;      FIRST ASCII CHARACTER ON IT
(1)  023372  012766  023404  000004            MOV     #$TTYIN,4(SP)
(1)  023400  000002                            RTI                             ;;RETURN
(1)  023402     000                    9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
(1)  023403     000                            .BYTE   0                       ;;TERMINATOR
(1)  023404  000010                    $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
(1)  023414  052536  005015     000    $CNTLU: .ASCIZ  /^U/<15><12>            ;;CONTROL "U"
(1)  023421     136  006507  000012    $CNTLG: .ASCIZ  /^G/<15><12>            ;;CONTROL "G"
(1)  023426  005015  053523  020122    $MSWR:  .ASCIZ  <15><12>/SWR = /
(1)  023434  020075     000
(1)  023437     040  047040  053505    $MNEW:  .ASCIZ  /  NEW = /
(1)  023444  036440  000040
2541                                   .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
(1)
(2)                                    ;;**************************************************************
(1)                                    ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
(1)                                    ;*CHANGE IT TO BINARY.
(1)                                    ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
(1)                                    ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
(1)                                    ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
(1)                                    ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
(1)                                    ;*CALL:
(1)                                    ;*      RDOCT                           ;;READ AN OCTAL NUMBER
(1)                                    ;*      RETURN HERE                     ;;LOW ORDER BITS ARE ON TOP OF THE STACK
(1)                                    ;*                                      ;;HIGH ORDER BITS ARE IN $HIOCT
(1)
(1)  023450  011646                    $RDOCT: MOV     (SP),-(SP)              ;;PROVIDE SPACE FOR THE
(1)  023452  016666  000004  000002            MOV     4(SP),2(SP)             ;;INPUT NUMBER
(3)  023460  010046                            MOV     R0,-(SP)                ;;PUSH R0 ON STACK
(3)  023462  010146                            MOV     R1,-(SP)                ;;PUSH R1 ON STACK
```

```
(3)   023464  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(1)   023466  104407          1$:      RDLIN                   ;;READ AN ASCIZ LINE
(1)   023470  012600                   MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
(1)   023472  010037  023576             MOV     R0,5$           ;;AND SAVE IT
(1)   023476  005001                   CLR     R1              ;;CLEAR DATA WORD
(1)   023500  005002                   CLR     R2
(1)   023502  112046          2$:      MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
(1)   023504  001420                   BEQ     3$              ;;IF ZERO GET OUT
(1)   023506  122716  000060             CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
(1)   023512  003026                   BGT     4$              ;;IS AN OCTAL DIGIT
(1)   023514  122716  000067             CMPB    #'7,(SP)
(1)   023520  002423                   BLT     4$
(1)   023522  006301                   ASL     R1              ;;*2
(1)   023524  006102                   ROL     R2
(1)   023526  006301                   ASL     R1              ;;*4
(1)   023530  006102                   ROL     R2
(1)   023532  006301                   ASL     R1              ;;*8
(1)   023534  006102                   ROL     R2
(1)   023536  042716  177770             BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
(1)   023542  062601                   ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
(1)   023544  000756                   BR      2$              ;;LOOP
(1)   023546  005726          3$:      TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
(1)   023550  010166  000012             MOV     R1,12(SP)       ;;SAVE THE RESULT
(1)   023554  010237  023606             MOV     R2,$HIOCT
(3)   023560  012602                   MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)   023562  012601                   MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)   023564  012600                   MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)   023566  000002                   RTI                     ;;RETURN
(1)   023570  005726          4$:      TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
(1)   023572  105010                   CLRB    (R0)            ;;SET A TERMINATOR
(1)   023574  104401                   TYPE                    ;;TYPE UP THRU THE BAD CHAR.
(1)   023576  000000          5$:      .WORD   0
(1)   023600  104401  001166             TYPE    ,$QUES          ;;"?" "CR" & "LF"
(1)   023604  000730                   BR      1$              ;;TRY AGAIN
(1)   023606  000000          $HIOCT:  .WORD   0               ;;HIGH ORDER BITS GO HERE
2542  023610  105777  155324   MSCOPE: TSTB    @SWR            ;TEST BIT 7
2543  023614  100003                   BPL     $SCOPE          ;NOT SET
2544  023616  005737  010650             TST     LOOP            ;TEST LOOP
2545  023622  001041                   BNE     $OVER           ;SET LOOP ON TEST
2546
2547                                   .SBTTL  SCOPE HANDLER ROUTINE
(1)
(2)                                    ;;*********************************************************************
(1)                                    ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
(1)                                    ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
(1)                                    ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
(1)                                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(1)                                    ;*SW14=1          LOOP ON TEST
(1)                                    ;*SW08=1          LOOP ON TEST IN SWR<7:0>
(1)                                    ;*CALL
(1)                                    ;*      SCOPE             ;;SCOPE=IOT
(1)
(1)   023624                           $SCOPE:
(1)   023624  032777  040000  155306   1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
(1)   023632  001035                   BNE     $OVER           ;;YES IF SW14=1
(1)                                    ;*****START OF CODE FOR THE XOR TESTER*****
```

```
(1)  023634  000416                    $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
(1)                                             ;                       ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
(1)  023636  013746  000004                    MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
(1)  023642  012737  023662  000004            MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
(1)  023650  005737  177060                    TST     @#177060        ;;TIME OUT ON XOR?
(1)  023654  012637  000004                    MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  023660  000414                            BR      $SVLAD          ;;GO TO THE NEXT TEST
(1)  023662  022626                    5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
(1)  023664  012637  000004                    MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  023670  000416                            BR      $OVER           ;;LOOP ON THE PRESENT TEST
(1)  023672                            6$:;#####END OF CODE FOR THE XOR TESTER#####
(1)  023672  032777  000400  155240            BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?    '
(1)  023700  001404                            BEQ     $SVLAD          ;;BR IF NO
(1)  023702  127737  155232  001102            CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
(1)  023710  001406                            BEQ     $OVER           ;;BR IF YES
(1)  023712  105237  001102            $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
(1)  023716  011637  001106                    MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
(1)  023722  105037  001103                    CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
(1)  023726  013777  001102  155206    $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
(1)  023734  013716  001106                    MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
(1)  023740  000002                            RTI                     ;;FIXES PS
2548
2549                                    .SBTTL  ERROR HANDLER ROUTINE
(1)
(2)                                     ;;**********************************************************************
(1)                                     ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
(1)                                     ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
(1)                                     ;*AND GO TO $ERRTYP ON ERROR
(1)                                     ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(1)                                     ;*SW15=1        HALT ON ERROR
(1)                                     ;*SW13=1        INHIBIT ERROR TYPEOUTS
(1)                                     ;*CALL
(1)                                     ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
(1)
(1)  023742                             $ERROR:
(2)  023742  113737  001102  001252            MOVB    $TSTNM,TSTNUM
(1)  023750  105237  001103            7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
(1)  023754  001775                            BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
(1)  023756  013777  001102  155156            MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
(1)  023764  005237  001112                    INC     $ERTTL          ;;INC THE ERROR COUNT
(1)  023770  011637  001116                    MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
(1)  023774  162737  000002  001116            SUB     #2,$ERRPC
(1)  024002  117737  155110  001114            MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
(1)  024010  032777  020000  155122            BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
(1)  024016  001004                            BNE     20$             ;;SKIP TYPEOUTS
(1)  024020  004737  024054                    JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
(1)  024024  104401  001167                    TYPE    ,$CRLF
(1)  024030                            20$:
(1)  024030  005777  155104            2$:     TST     @SWR            ;;HALT ON ERROR
(1)  024034  100001                            BPL     3$              ;;SKIP IF CONTINUE
(1)  024036  000000                            HALT                    ;;HALT ON ERROR!
(1)  024040                            3$:
(1)  024040  022737  007014  000042            CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
(1)  024046  001001                            BNE     6$              ;;BRANCH IF NO
(1)  024050  000000                            HALT                    ;;YES
(1)  024052                            6$:
```

```
                                        RTI                             ;;RETURN

                                .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                                ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
                                ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

024054                          $ERRTYP:
024054  104401  001167                  TYPE    ,$CRLF                  ;;"CARRIAGE RETURN" & "LINE FEED"
024060  010046                          MOV     RO,-(SP)                ;;SAVE RO
024062  005000                          CLR     RO                      ;;PICKUP THE ITEM INDEX
024064  153700  001114                  BISB    @#$ITEMB,RO
024070  001004                          BNE     1$                      ;;IF ITEM NUMBER IS ZERO, JUST
                                                                        ;;TYPE THE PC OF THE ERROR
024072  013746  001116                  MOV     $ERRPC,-(SP)            ;;SAVE $ERRPC FOR TYPEOUT
                                                                        ;;ERROR ADDRESS
024076  104402                          TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
024100  000426                          BR      6$                      ;;GET OUT
024102  005300                  1$:     DEC     RO                      ;;ADJUST THE INDEX SO THAT IT WILL
024104  006300                          ASL     RO                      ;;      WORK FOR THE ERROR TABLE
024106  006300                          ASL     RO
024110  006300                          ASL     RO
024112  062700  001172                  ADD     #$ERRTB,RO              ;;FORM TABLE POINTER
024116  012037  024126                  MOV     (RO)+,2$                ;;PICKUP "ERROR MESSAGE" POINTER
024122  001404                          BEQ     3$                      ;;SKIP TYPEOUT IF NO POINTER
024124  104401                          TYPE                            ;;TYPE THE "ERROR MESSAGE"
024126  000000                  2$:     .WORD   0                       ;;"ERROR MESSAGE" POINTER GOES HERE
024130  104401  001167                  TYPE    ,$CRLF                  ;;"CARRIAGE RETURN" & "LINE FEED"
024134  012037  024144          3$:     MOV     @(RO)+,4$               ;;PICKUP "DATA HEADER" POINTER
024140  001404                          BEQ     5$                      ;;SKIP TYPEOUT IF 0
024142  104401                          TYPE                            ;;TYPE THE "DATA HEADER"
024144  000000                  4$:     .WORD   0                       ;;"DATA HEADER" POINTER GOES HERE
024146  104401  001167                  TYPE    ,$CRLF                  ;;"CARRIAGE RETURN" & "LINE FEED"
024152  011000                  5$:     MOV     @(RO),RO                ;;PICKUP "DATA TABLE" POINTER
024154  001004                          BNE     7$                      ;;GO TYPE THE DATA
024156  012600                  6$:     MOV     (SP)+,RO                ;;RESTORE RO
024160  104401  001167                  TYPE    ,$CRLF                  ;;"CARRIAGE RETURN" & "LINE FEED"
024164  000207                          RTS     PC                      ;;RETURN
024166                          7$:
024166  013046                          MOV     @(RO)+,-(SP)            ;;SAVE @(RO)+ FOR TYPEOUT
024170  104402                          TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
024172  005710                          TST     (RO)                    ;;IS THERE ANOTHER NUMBER?
024174  001770                          BEQ     6$                      ;;BR IF NO
024176  104401  024204                  TYPE    ,8$                     ;;TYPE TWO(2) SPACES
024202  000771                          BR      7$                      ;;LOOP
024204  020040      000         8$:     .ASCIZ  /  /                    ;;TWO(2) SPACES
024210                                  .EVEN

                                .SBTTL  TYPE ROUTINE

                                ;;**********************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
```

```
(1)                                     ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
(1)                                     ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
(1)                                     ;*
(1)                                     ;*CALL:
(1)                                     ;*1) USING A TRAP INSTRUCTION
(1)                                     ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
(1)                                     ;*OR
(1)                                     ;*      TYPE
(1)                                     ;*      MESADR
(1)                                     ;*
(1)
(1)  024210  105737  001157     $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
(1)  024214  100002                     BPL     1$              ;;BR IF YES
(1)  024216  000000                     HALT                    ;;HALT HERE IF NO TERMINAL
(1)  024220  000407                     BR      3$              ;;LEAVE
(1)  024222  010046             1$:     MOV     R0,-(SP)        ;;SAVE R0
(1)  024224  017600  000002             MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
(1)  024230  112046             2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(1)  024232  001005                     BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(1)  024234  005726                     TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(1)  024236  012600             60$:    MOV     (SP)+,R0        ;;RESTORE R0
(1)  024240  062716  000002     3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
(1)  024244  000002                     RTI                     ;;RETURN
(1)  024246  122716  000011     4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(1)  024252  001430                     BEQ     8$
(1)  024254  122716  000200             CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(1)  024260  001006                     BNE     5$
(1)  024262  005726                     TST     (SP)+           ;;POP  <CR><LF> EQUIV
(1)  024264  104401                     TYPE                    ;;TYPE A CR AND LF
(1)  024266  001167                     $CRLF
(1)  024270  105037  024424             CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
(1)  024274  000755                     BR      2$              ;;GET NEXT CHARACTER
(1)  024276  004737  024360     5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(1)  024302  123726  001156     6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(1)  024306  001350                     BNE     2$              ;;IF NO GO GET NEXT CHAR.
(1)  024310  013746  001154             MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(1)                                                             ;;AND THE NULL CHAR.
(1)  024314  105366  000001     7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(1)  024320  002770                     BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1)  024322  004737  024360             JSR     PC,$TYPEC       ;;GO TYPE A NULL
(1)  024326  105337  024424             DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(1)  024332  000770                     BR      7$              ;;LOOP
(1)
(1)                                     ;HORIZONTAL TAB PROCESSOR
(1)
(1)  024334  112716  000040     8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(1)  024340  004737  024360     9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
(1)  024344  132737  000007  024424     BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(1)  024352  001372                     BNE     9$              ;;TAB STOP
(1)  024354  005726                     TST     (SP)+           ;;POP SPACE OFF STACK
(1)  024356  000724                     BR      2$              ;;GET NEXT CHARACTER
(1)  024360  105777  154564     $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(1)  024364  100375                     BPL     $TYPEC
(1)  024366  116677  000002  154556     MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)  024374  122766  000015  000002     CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(1)  024402  001003                     BNE     1$              ;;BRANCH IF NO
```

```
(1)  024404  105037  024424                    CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(1)  024410  000406                            BR      $TYPEX          ;;EXIT
(1)  024412  122766  000012  000002  1$:      CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(1)  024420  001402                            BEQ     $TYPEX          ;;BRANCH IF YES
(1)  024422  105227                            INCB    (PC)+           ;;COUNT THE CHARACTER
(1)  024424  000000                    $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
(1)  024426  000207                    $TYPEX: RTS     PC
(1)
2554
2555                                    .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
(1)
(2)                                     ;;**************************************************************
(1)                                     ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
(1)                                     ;*OCTAL (ASCII) NUMBER AND TYPE IT.
(1)                                     ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
(1)                                     ;*CALL:
(1)                                     ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                     ;*      TYPOS                   ;;CALL FOR TYPEOUT
(1)                                     ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
(1)                                     ;*      .BYTE   M               ;;M=1 OR 0
(1)                                     ;*                                      ;;1=TYPE LEADING ZEROS
(1)                                     ;*                                      ;;0=SUPPRESS LEADING ZEROS
(1)                                     ;*
(1)                                     ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
(1)                                     ;*$TYPOS OR $TYPOC
(1)                                     ;*CALL:
(1)                                     ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                     ;*      TYPON                   ;;CALL FOR TYPEOUT
(1)                                     ;*
(1)                                     ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
(1)                                     ;*CALL:
(1)                                     ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                     ;*      TYPOC                   ;;CALL FOR TYPEOUT
(1)
(1)  024430  017646  000000            $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
(1)  024434  116637  000001  024653            MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
(1)  024442  112637  024655                    MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
(1)  024446  062716  000002                    ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
(1)  024452  000406                            BR      $TYPON
(1)  024454  112737  000001  024653    $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
(1)  024462  112737  000006  024655            MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
(1)  024470  112737  000005  024652    $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
(1)  024476  010346                            MOV     R3,-(SP)        ;;SAVE R3
(1)  024500  010446                            MOV     R4,-(SP)        ;;SAVE R4
(1)  024502  010546                            MOV     R5,-(SP)        ;;SAVE R5
(1)  024504  113704  024655                    MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
(1)  024510  005404                            NEG     R4
(1)  024512  062704  000006                    ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
(1)  024516  110437  024654                    MOVB    R4,$OMODE       ;;SAVE IT FOR USE
(1)  024522  113704  024653                    MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
(1)  024526  016605  000012                    MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
(1)  024532  005003                            CLR     R3              ;;CLEAR THE OUTPUT WORD
(1)  024534  006105                    1$:     ROL     R5              ;;ROTATE MSB INTO "C"
(1)  024536  000404                            BR      3$              ;;GO DO MSB
(1)  024540  006105                    2$:     ROL     R5              ;;FORM THIS DIGIT
(1)  024542  006105                            ROL     R5
```

```
(1)   024544  006105                        ROL     R5
(1)   024546  010503                        MOV     R5,R3
(1)   024550  006103                3$:     ROL     R3                      ;;GET LSB OF THIS DIGIT
(1)   024552  105337  024654                DECB    $OMODE                  ;;TYPE THIS DIGIT?
(1)   024556  100016                        BPL     7$                      ;;BR IF NO
(1)   024560  042703  177770                BIC     #177770,R3              ;;GET RID OF JUNK
(1)   024564  001002                        BNE     4$                      ;;TEST FOR 0
(1)   024566  005704                        TST     R4                      ;;SUPPRESS THIS 0?
(1)   024570  001403                        BEQ     5$                      ;;BR IF YES
(1)   024572  005204                4$:     INC     R4                      ;;DON'T SUPPRESS ANYMORE 0'S
(1)   024574  052703  000060                BIS     #'0,R3                  ;;MAKE THIS DIGIT ASCII
(1)   024600  052703  000040          5$:   BIS     #' ,R3                  ;;MAKE ASCII IF NOT ALREADY
(1)   024604  110337  024650                MOVB    R3,8$                   ;;SAVE FOR TYPING
(1)   024610  104401  024650                TYPE    8$                      ;;GO TYPE THIS DIGIT
(1)   024614  105337  024652          7$:   DECB    $OCNT                   ;;COUNT BY 1
(1)   024620  003347                        BGT     2$                      ;;BR IF MORE TO DO
(1)   024622  002402                        BLT     6$                      ;;BR IF DONE
(1)   024624  005204                        INC     R4                      ;;INSURE LAST DIGIT ISN'T A BLANK
(1)   024626  000744                        BR      2$                      ;;GO DO THE LAST DIGIT
(1)   024630  012605                6$:     MOV     (SP)+,R5                ;;RESTORE R5
(1)   024632  012604                        MOV     (SP)+,R4                ;;RESTORE R4
(1)   024634  012603                        MOV     (SP)+,R3                ;;RESTORE R3
(1)   024636  016666  000002  000004        MOV     2(SP),4(SP)             ;;SET THE STACK FOR RETURNING
(1)   024644  012616                        MOV     (SP)+,(SP)
(1)   024646  000002                        RTI                             ;;RETURN
(1)   024650      000                8$:     .BYTE   0                       ;;STORAGE FOR ASCII DIGIT
(1)   024651      000                        .BYTE   0                       ;;TERMINATOR FOR TYPE ROUTINE
(1)   024652      000                $OCNT:  .BYTE   0                       ;;OCTAL DIGIT COUNTER
(1)   024653      000                $OFILL: .BYTE   0                       ;;ZERO FILL SWITCH
(1)   024654  000000                 $OMODE: .WORD   0                       ;;NUMBER OF DIGITS TO TYPE
2556                                .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
(1)
(2)                                 ;;**************************************************************
(1)                                 ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
(1)                                 ;*WITH A RANGE OF 0 TO 2(+33)-1.
(1)                                 ;*CALL:
(1)                                 ;*      JSR     PC,$RAND                ;;CALL THE ROUTINE
(1)                                 ;*      RETURN                          ;;RETURN HERE THE RANDOM
(1)                                 ;*                                      ;;NUMBER WILL BE IN
(1)                                 ;*                                      ;;$HINUM,$LONUM
(1)
(1)   024656                        $RAND:
(3)   024656  010046                        MOV     R0,-(SP)                ;;PUSH R0 ON STACK
(3)   024660  010146                        MOV     R1,-(SP)                ;;PUSH R1 ON STACK
(3)   024662  010246                        MOV     R2,-(SP)                ;;PUSH R2 ON STACK
(1)   024664  013700  024756                MOV     $LONUM,R0               ;;SET R0 WITH LOW
(1)   024670  013701  024754                MOV     $HINUM,R1               ;;SET R1 WITH HIGH
(1)   024674  012702  177771                MOV     #-7,R2                  ;;SET SHIFT COUNT
(1)   024700  006300                1$:     ASL     R0                      ;;SHIFT R0 LEFT AND
(1)   024702  006101                        ROL     R1                      ;;ROTATE CARRY INTO R1 AND
(1)   024704  005202                        INC     R2                      ;;CHECK FOR DONE
(1)   024706  001374                        BNE     1$                      ;;CONTINUE SHIFT LOOP
(1)   024710  063700  024756                ADD     $LONUM,R0               ;;ADD NUMBER TO MAKE X 129
(1)   024714  005501                        ADC     R1                      ;;PROPOGATE CARRY
(1)   024716  063701  024754                ADD     $HINUM,R1               ;;ADD NUMBER TO MAKE X 129
(1)   024722  062700  001057                ADD     #1057,R0                ;;ADD LOW CONSTANT
```

```
(1)  024726  005501                    ADC     R1              ;;PROPOGATE CARRY
(1)  024730  062701  047401            ADD     #47401,R1       ;;ADD HIGH CONSTANT
(1)  024734  010037  024756            MOV     R0,$LONUM       ;;SAVE R0
(1)  024740  010137  024754            MOV     R1,$HINUM       ;;SAVE R1
(3)  024744  012602                    MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  024746  012601                    MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  024750  012600                    MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  024752  000207                    RTS     PC              ;;RETURN
(1)  024754  176543            $HINUM: .WORD   176543
(1)  024756  123456            $LONUM: .WORD   123456
2557                           .SBTTL  TRAP DECODER
(1)
(2)                            ;;**************************************************************
(1)                            ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
(1)                            ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
(1)                            ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
(1)                            ;*GO TO THAT ROUTINE.
(1)
(1)  024760  010046            $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
(1)  024762  016600  000002            MOV     2(SP),R0        ;;GET TRAP ADDRESS
(1)  024766  005740                    TST     -(R0)           ;;BACKUP BY 2
(1)  024770  111000                    MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
(1)  024772  006300                    ASL     R0              ;;POSITION FOR INDEXING
(1)  024774  016000  025014            MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
(1)  025000  000200                    RTS     R0              ;;GO TO ROUTINE
(1)
(1)
(1)                            ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
(1)
(1)  025002  011646            $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
(1)  025004  016666  000004  000002    MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
(1)  025012  000002                    RTI                     ;;RESTORE THE PSW
(1)
(3)                            .SBTTL  TRAP TABLE
(3)
(3)                            ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
(3)                            ;*BY THE "TRAP" INSTRUCTION.
(3)
(3)                            ;       ROUTINE
(3)                            ;       -------
(3)  025014  025002            $TRPAD: .WORD   $TRAP2
(3)  025016  024210                    $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
(3)  025020  024454                    $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
(3)  025022  024430                    $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
(3)  025024  024470                    $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
(3)  025026  012564                    $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
(1)
(1)
(3)  025030  023156                    $RDCHR  ;;CALL=RDCHR    TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
(3)  025032  023276                    $RDLIN  ;;CALL=RDLIN    TRAP+7(104407)  TTY TYPEIN STRING ROUTINE
(3)  025034  023450                    $RDOCT  ;;CALL=RDOCT    TRAP+10(104410) READ AN OCTAL NUMBER FROM TTY
2558                           .SBTTL  POWER DOWN AND UP ROUTINES
(1)
(2)                            ;;**************************************************************
(1)                            ;POWER DOWN ROUTINE
(1)  025036  012737  025176  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
```

```
(1)  025044  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
(3)  025052  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  025054  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  025056  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  025060  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  025062  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
(3)  025064  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(3)  025066  017746  154046                  MOV     @#SWR,-(SP)     ;;PUSH @#SWR ON STACK
(1)  025072  010637  025202                  MOV     SP,$SAVR6       ;;SAVE SP
(1)  025076  012737  025110  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
(1)  025104  000000                          HALT
(1)  025106  000776                          BR      .-2             ;;HANG UP
(1)
(2)                                  ;;**************************************************************
(1)                                  ;POWER UP ROUTINE
(1)  025110  012737  025176  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
(1)  025116  013706  025202                  MOV     $SAVR6,SP       ;;GET SP
(1)  025122  005037  025203                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
(1)  025126  005237  025202          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
(1)  025132  001375                          BNE     1$              ;;OF WORD
(3)  025134  012677  154000                  MOV     (SP)+,@#SWR     ;;POP STACK INTO @#SWR
(3)  025140  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  025142  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
(3)  025144  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  025146  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  025150  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  025152  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  025154  012737  025036  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(1)  025162  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
(1)  025170  104401                          TYPE                    ;REPORT THE POWER FAILURE
(1)  025172  025204          $PWRMG: .WORD   $POWER                  ;;POWER FAIL MESSAGE POINTER
(1)  025174  000002                  RTI
(1)  025176  000000          $ILLUP: HALT                            ;;THE POWER UP SEQUENCE WAS STARTED
(1)  025200  000776                  BR      .-2                     ;;  BEFORE THE POWER DOWN WAS COMPLETE
(1)  025202  000000          $SAVR6: 0                               ;;PUT THE SP HERE
(1)  025204  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
(1)  025212  000122
(1)                                  .EVEN
2562  025214  000010                HOLD:   .BLKW   10
2563  025234  000010                GOOD:   .BLKW   10
2564  025254  000000                RETRY:  .WORD   0
2565  025256  000000                BUFFER: 0
2566          000001                        .END
```

```
A          007432
ADELAY     011370
AKPTAB     021660
AMSG       011442
ANESC      010664
AVAIS      022014
AXOFF      010656
B          007774
BEGIN      001334
BEGIN1     001410
BEGIN2     001420
BEGIN3     001400
BFCHAR     007336
BFCHR      007276
BIT0    =  000001
BIT00   =  000001
BIT01   =  000002
BIT02   =  000004
BIT03   =  000010
BIT04   =  000020
BIT05   =  000040
BIT06   =  000100
BIT07   =  000200
BIT08   =  000400
BIT09   =  001000
BIT1    =  000002
BIT10   =  002000
BIT11   =  004000
BIT12   =  010000
BIT13   =  020000
BIT14   =  040000
BIT15   =  100000
BIT2    =  000004
BIT3    =  000010
BIT4    =  000020
BIT5    =  000040
BIT6    =  000100
BIT7    =  000200
BIT8    =  000400
BIT9    =  001000
BPTVEC=    000014
BUFFER     025256
C          007510
COPTST     005362
CR      =  000015
CRLF    =  000200
CRLFA      016474
CRLFB      016463
D          007576
DCATST     004462
DDISP   =  177570
DELAY      011262
DH1        020570
DH3        020617
DH4        020666
DIG0       011532
DIG1       011534
DIG2       011556
DISPCH     006616
DISPLA     001142
DISPRE     000174
DSPCH      010776
DSWR    =  177570
DTPSR      011154
DTPSRA     011202
DTPSRB     011160
DT1        020734
DT3        020744
DT4        020760
E          007646
EMTVEC=    000030
EM1        020345
EM2        020412
EM3        020441
EM4        020466
EM5        020530
ENAKP      021543
ENGRAF     021555
ENPNTM     021567
ERCHR      022025
ERRVEC=    000004
EXAKP      021550
EXGRAF     021562
EXPNTM     021574
F          007716
FILBFA     010154
FILBFB     010140
FILBUF     010134
FILLWC     013010
FINDOT     010676
FINDTA     010706
FIRLST     011060
FIRST      001242
FNDA       010670
GBBUF      011220
GENER      004546
GENRX      004632
GETCHR     013052
GIN        001426
GINA       001434
GOAPMD     020054
GODSHS     020026
GOHDSC     020000
GONAPM     020061
GOOD       025234
GOPRNT     017752
GRPHST     005266
HOLD       025214
HOMERS     016503
HT      =  000011
IGNORE     010666
INSTR      022046
INVLID     016134
IOTVEC=    000020
KPTAB      021602
KRBDON     010064
KRBECH     010100
KRBECO     007110
KRBTST     007404
LAST       001244
LASTCH     001310
LASTLN     001262
LBMT       002774
LBMT1      003144
LCM        003154
LF      =  000012
LIC        010170
LICA       010206
LICB       010224
LODERL     004016
LODERS     004174
LOOP       010650
LRL        003436
LTAB       003706
MANFU      001360
MASK1      011766
MASK2      011770
MKB        014321
MKBA       014410
MKBB       014477
MKBB2      014614
MKBC       014677
MKBD       014736
MKBD2      015043
MKBE       015131
MKBF       015251
MKBG       015307
MKBGA      015356
MKBH       015426
MKBI       015467
MKBJ       015522
MKBK       015555
MKBL       015611
MKBM       015644
MKBN       015700
MKBQ       015773
MKBQA      016040
MKBQB      016057
MKBQ1      016034
MKBQ2      016053
MKBR       016066
MKBS1      014353
MKBS2      015727
MKE        016202
MKEA       016407
MKEA1      016420
MKEB       016425
MKEH       016434
MOVDN1     003110
MOVDWN     003104
MOVRIG     003126
MPTCNT     014160
MPTSCN     014227
MQ0        017556
MQ1        017643
MQ2        017731
MSCOPE     023610
MSGTNO     005076
MSGTXT     004756
MSGTYP     002436
MTEXT0     022131
MTEXT1     022235
MTEXT2     022336
MTEXT3     022440
MTEXT4     022523
MTEXT5     022625
MTEXT6     022726
MTEXT7     022760
MTEXT8     023060
MTOB       013130
M91        013245
M910       013536
M911       013567
M912       013622
M914       013656
M92        013313
M920       013723
M921       013753
M922       014007
M9221      014044
M923       014121
M923A      014264
M93        013342
M94        013404
M96        013435
M97        013470
M98        020110
NOWEOP     007050
OCTAL      011462
OUT        021777
OUTPUT     012400
OVRAL      004754
PARITY     011772
PASHED     020300
PATH       016511
PIRQ    =  177772
PIRQVE=    000240
PNTWID     001316
PRNTST     006342
PRTCNT     001274
PR0     =  000000
PR1     =  000040
PR2     =  000100
PR3     =  000140
PR4     =  000200
PR5     =  000240
PR6     =  000300
PR7     =  000340
PS      =  177776
PSW     =  177776
PTCT       001250
PWRVEC=    000024
RBEGIN     001364
RDCHR   =  104406
RDLIN   =  104407
RDOCT   =  104410
RESVEC=    000010
RETRY      025254
REVLF      021536
RFI        017551
ROW1       021030
ROW1C      021444
ROW1S      021352
ROW12      021200
ROW12C     021500
ROW12S     021406
ROW2       021064
ROW22      021236
ROW3       021122
ROW32      021274
ROW4       021152
ROW42      021326
ROW5       021176
RSTRT      002014
R6      =%000006
R7      =%000007
SAVE1      001322
SAVE2      001324
SAVE3      001326
SAVE4      001330
SBEGIN     002006
SET        004752
SHBE       021766
STACK   =  001100
STCHAR     001306
```

```
STKLMT= 177774
SUBTST  001256
SWR     001140
SWREG   000176
SW0   = 000001
SW00  = 000001
SW01  = 000002
SW02  = 000004
SW03  = 000010
SW04  = 000020
SW05  = 000040
SW06  = 000100
SW07  = 000200
SW08  = 000400
SW09  = 001000
SW1   = 000002
SW10  = 002000
SW11  = 004000
SW12  = 010000
SW13  = 020000
SW14  = 040000
SW15  = 100000
SW2   = 000004
SW3   = 000010
SW4   = 000020
SW5   = 000040
SW6   = 000100
SW7   = 000200
SW8   = 000400
SW9   = 001000
TBITVE= 000014
TEMP    001312
TEMPO   001314
TIME0   001254
TIME1   013124
TIME2   013126
TITLE   013144
TKVEC = 000060
TOTALC  001264
TPVEC = 000064
TRAPVE= 000034
TRTVEC= 000014
TSTDAT  012270
TSTKPS  012164
TSTNUM  001252
TSTROW  011560
TST1    002042
TST10   003776
TST11   004154
TST12   004302
TST13   004340
TST14   005100
TST15   005250
TST16   005362
TST17   005532
TST2    002466
TST20   005566
TST21   005656
TST22   006010
TST23   006144
TST24   006266
TST25   006344
TST26   006456
TST27   006512
TST3    002506
TST30   007106
TST31   007402
TST32   010076
TST4    002562
TST5    002662
TST6    002764
TST7    003426
TYPDS = 104405
TYPE  = 104401
TYPEPT  002404
TYPOC = 104402
TYPON = 104404
TYPOS = 104403
VH0     001266
VH1     001270
VH2     001272
VTIB    001300
VTIS    001276
VTNOW   001246
VT0B    001304
VT0S    001302
VT5XX   001260
VT50A   016760
VT50B   017032
VT50H   017163
VT52K   017241
VT52L   017324
VT52M   017405
VT55    017101
VT55E   017466
VSORW   020774
VS2RW   021012
WFTEST  001332
WHAT0   020156
WHAT1   020221
WHERE   002402
WIDTH   001320
XADDS   004750
XOFF  = 000023
XOFFBR  010652
XOFFOK  010654
XOFFRC  010660
XON   = 000021
XONRC   010662
XPRNT   010236
XPRNTA  010242
XPRNTB  010260
YADDS   004746
$AUTOB  001134
$BDADR  001122
$BDDAT  001126
$CHARC  024424
$CMTAG  001100
$CM1  = 000002
$CM2  = 000004
$CM3  = 000002
$CNTLG  023421
$CNTLU  023414
$CRLF   001167
$DBLK   013000
$DOAGN  007024
$DTBL   012770
$ENDAD  007014
$ENDCT  006762
$ENDMG  007033
$ENULL  007030
$EOP    006644
$EOPCT  006754
$ERFLG  001103
$ERMAX  001115
$ERROR  023742
$ERRPC  001116
$ERRTB  001172
$ERRTY  024054
$ERTTL  001112
$FILLC  001156
$FILLS  001155
$GDADR  001120
$GDDAT  001124
$GET4   007044
$HO   = 000000
$HINUM  024754
$HIOCT  023606
$ICNT   001104
$ILLUP  025176
$INTAG  001135
$ITEMB  001144
$LF     001170
$LONUM  024756
$LPADR  001106
$LPERR  001110
$MNEW   023437
$MSWR   023426
$NULL   001154
$NWTST= 000001
$OCNT   024652
$OMODE  024654
$OVER   023726
$PASS   001100
$POWER  025204
$PWRDN  025036
$PWRMG  025172
$PWRUP  025110
$QUES   001166
$RAND   024656
$RDCHR  023156
$RDLIN  023276
$RDOCT  023450
$RDSZ   = 000010
$REGAD  001160
$REGO   001162
$REGI   001164
$RTNAD  007026
$SAVR6  025202
$SCOPE  023624
$SETUP= 000037
$STUP   = 177777
$SVLAD  023712
$SWR    = 160400
$SWRMK= 000000
$TKB    001146
$TKS    001144
$TN     = 000033
$TPB    001152
$TPFLG  001157
$TPS    001150
$TRAP   024760
$TRAP2  025002
$TRP    = 000011
$TRPAD  025014
$TSTNM  001102
$TTYIN  023404
$TYPDS  012564
$TYPE   024210
$TYPEC  024360
$TYPEX  024426
$TYPOC  024454
$TYPON  024470
$TYPOS  024430
$XTSTR  023634
$$GET4= 000000
$OFILL  024653
.       = 025260
. ABS.  025260      000

 ERRORS DETECTED:  0

 DZVTCE,DZVTCE.SEQ=DZVTCE
 RUN-TIME: 20 10  .5 SECONDS
 RUN-TIME RATIO: 457/31=14.3
 CORE USED:  24K  (47 PAGES)
```

E06